Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electric Power Engineering

Viljar Fjellanger

On-line Voltage Estimation during Sensorless Control of an Induction Machine Drive

Master's thesis in Electric Power Engineering
Supervisor: Roy Nilsen
June 2019

NTNU
Norwegian University of Science and Technology

Viljar Fjellanger

On-line Voltage Estimation during Sensorless Control of an Induction Machine Drive

Master's thesis in Electric Power Engineering
Supervisor: Roy Nilsen
June 2019

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Electric Power Engineering

NTNU
Kunnskap for en bedre verden

Problem Description

In the research group Power Electronics System and Components (PESC) at the department of Electric Power Engineering a new Control-platform is developed. This control platform is based on a picoZed -board with a Xilinx Zynq7030 System-on-Chip (SoC) with two floating point processors and a Field Programmable Gate Array (FPGA). The tools used are Vivado for FPGA programming, System Generator and Simulink for developing IP-cores for the FPGA and finally SDK for C++ programming.

One of the applications of this Control-platform is Sensorless control of Electric Machines. This thesis shall be focused on a 3-phase Induction Motor drive. The tasks to be performed are:

- Derive mathematical models and present the control theory for Induction Motor Drives.
- Discuss the most commonly used flux models: Current Model and Voltage Model.
- Evaluate the sources of error in these models; especially voltage errors.
- Present the theory for the effect on dead-times and blanking times on in the pulse pattern in a 2-level 3-phase inverter.
- Develop a voltage-estimator to be used in a flux model. The estimator should be implemented in the FPGA by help of Simulink and the System Generator package from Xilinx.
- Methods and Workflow for how to test this in Simulink and on the Control Platform should be presented if sufficient time is available.

Supervisor: Prof. Roy Nilsen

Preface

This thesis was submitted to the Department of Electric Power Engineering at NTNU and concludes the 2-year master program “Electric Power Engineering”. The work has been challenging and motivating and really widened my horizon in the exiting world of programming. I am very grateful for the new acquired knowledge.

I want to thank my supervisor Prof. Roy Nilsen for all his guidance throughout the project. His theoretical knowledge, constructive feedback and passion for the field of study has been very valuable during this work. I would also give a special thanks to PhD candidate Anirudh Budnar Acharya for sharing his knowledge in FPGA programming and his invaluable assistance with the FPGA design. Lastly, I want to thank the participants in the PESC group for all the interesting and fruitful discussions.

Trondheim, June 11, 2019

Viljar Fjellanger

Abstract

Sensorless control of electrical machines can increase reliability and reduce cost of high-performance drives. The sensorless control can be achieved by using *field-oriented control*, where voltage and current measurement are used as inputs to a flux model called *voltage model*. However, it is well known that errors in input variables, such as stator voltage, can lead to *drifting* in the flux linkage estimation. It is possible to give an estimate of the stator voltage using control signal for the inverter's IGBTs and correcting for turn-on and turn-off delays of the IGBTs and blanking time.

The main purpose of this thesis was to design and test an on-line voltage estimator using Xilinx's System Generator DSP tool together with Xilinx SDK. The voltage estimator was tested using an open loop V/f-controller with the future goal of implementing it in a field-oriented control. Tests were performed purely on the control board without a physical inverter connected. Test results were compared to simulations and the expected results deducted in the theoretical study. Modelling, scaling and control strategies of the induction machine was thoroughly reviewed with updated concepts and notations and a non-ideal inverter model was presented and analysed in order to design the voltage estimator.

The voltage estimator used the CPU for mathematical computations and the FPGA to estimate the time period to which the inverter outputs had same voltage potential as DC-link voltage. Theoretical analysis showed that blanking time and difference in turn-on and turn-off time in the IGBTs introduced a time delay to the output voltage pulse, compared to control signal of the IGBT. It was also shown that the time delay was dependent on current direction. The voltage estimator's ability to introduce this time delay was initially tested. The test was performed by using one bridge leg as a half-bridge converter and monitoring the on-time counter value. Results from tests with a constant control voltage and a switch in current direction, showed correct time delays in on-time estimation.

Phase voltage estimation was tested during low speed V/f-control. The test was performed with and without blanking time. Results without blanking time showed an estimated phase voltage equal to the control voltage and test with blanking time showed estimation of a distorted output voltage. The distorted waveform of estimated output voltage was in accordance with the theoretical analysis.

An expression for the non-linear voltage drop imposed by the inverter was deducted in the non-ideal inverter model. The expression was used to create a correction to the control voltage with the intention to remove the distortion of output voltage. The estimator was tested similarly as for the initial phase-voltage test. The result showed an estimated output voltage equal to the sinusoidal reference voltage from the controller. From this it could be verified that the non-linear voltage drop equation given by the inverter model was correct. Furthermore, similarities between results from the compensated V/f-control and the expected results from a FOC, showed that the estimator could work independently of control strategy.

From the results it was concluded that the new IP-core that was integrated in the FPGA design was able to recognise current direction and insert the correct time delay to the on-time count. It was also concluded that voltage estimator was able to estimate correct output voltage, independent of control strategy. This verified the equations for output voltage and distorted voltage deducted in non-ideal inverter model.

Finally, Xilinx System Generator DPS tool was proven to an efficient way of programming complex FPGA programs. The tool allowed for design, testing and implementation of new IP-cores without knowledge of VHDL programming.

Sammendrag

Sensorløs kontroll av elektriske maskiner kan bidra til økt pålitelighet samt kostnadsreduksjon på motordrifter med krav til høy-presisjon. Kontrollstrategien *feltorientert kontroll* kan brukes for å oppnå sensorløs kontroll ved at strøm- og spenningsmålinger brukes i en fluksmodell som kalles *spennings-modellen*. Det er videre et kjent problem at unøyaktige inputvariabler i spenningsmodellen, som for eksempel spenningsmålinger, kan føre til feil i fluksestimeringen. I litteraturen kalles denne feilen for *drifting*. Det er mulig å estimere statorspenningen ved hjelp av styresignalene til IGBTene i omformeren. Ved slik estimering er det viktig å korrigere for tidsforsinkelser i forbindelse med svitsjeforløpet til IGBTene samt dødtid i omformeren.

Hovedformålet med oppgaven har vært design og testing av en on-line spenningsestimator. Estimatoren er designet ved hjelp programmene System Generator DSP og SDK, begge utviklet og laget av Xilinx. Spenningsestimatoren er testet ved hjelp av en åpen sløyfe V/f-kontroller med et framtidig mål om å implementere den i en feltorientert kontroller. Testene i denne masteroppgaven ble gjort på et kontrollbrett uten tilkobling til en fysisk omformer. Resultatene ble sammenlignet med simuleringer og sett i sammenheng med de forventede resultatene fra teoristudiet. Modellering, skalering og kontrollstrategier for induksjonsmaskinen ble nøye gjennomgått. Likningene ble presentert med oppdatert notasjon i henhold til motordrift faget på fakultetet. En ikke-ideell omformermodell ble også presentert og analysert for å forstå hvordan spenningsestimatoren skulle designes.

Matematiske beregninger til estimatoren ble gjort i prosessoren mens estimatet av tidsintervallet, hvor spenningen på omformerens utgangsterminaler var lik DC-spenningen, ble gjort i FPGAen. Analysen av omformeren viste at differansen i av og på forsinkelsen i IGBTene samt dødtiden, introduserte en tidsforsinkelse i spenningspulsen fra omformeren i forhold til kontrollsignalet til IGBTen. Det ble også vist at tidsforsinkelsen var avhengig av strømretningen i brogrenen. For å teste FPGA-programmets evne til å estimere riktig tidsintervall, ble den ene brogrenen brukt som en halvbro-omformer og påtrykt en konstant kontrollspenning. Strømretningen ble også endret under testen. Testresultatene viste at estimatoren var i stand til å gi korrekte resultater.

Fasespenningsestimat ble testet ved bruk av en V/f-kontroller innstilt for lav utgangshastighet. Testen ble gjennomført med og uten dødtid. Resultatet fra testen uten dødtid vise et fasespenningsestimat likt kontrollspenningen. Resultatet med dødtid viste en forvrenging i den estimerte utgangsspenningen, helt i samsvar med teoristudiet.

Et utrykk for det ulineære spenningsfallet fra omformeren ble utledet i den ikke-ideelle omformer-modellen. Uttrykket ble brukt til å korrigere kontrollspenningen med mål om fjerne det ulineære spenningsfallet. Testresultatene viste at den estimerte utgangsspenningen hadde samme fase og amplitude som den sinusformede referansespenningen fra omformeren. Resultatet viste at det utledede utrykket som beskriver spenningsfallet var korrekt. Resultatene fra testen var også lik det man kunne forventet seg ved bruk av feltorientert kontroll. Dette viste at estimator ikke var begrenset til en kontroll-strategi.

Fra resultatene konkluderes det at den nye IP-kjernen som ble integrert i det ferdige FPGA-designet, evnet å estimere riktig tidsintervall for utgangsspenningen. Det ble også konkludert at spennings-estimatoren klarte å estimere riktig utgangsspenning i de forskjellige testene og at estimering var uavhengig av kontrollstrategien. Dette ga grunnlaget for verifisering av likningene som ble funnet i den ikke-ideelle omformermodellen.

Xilinx System Generator DPS-verktøy har vist seg å være en svært effektiv måte å programmere komplekse FPGA-programmer på. Verktøyet muliggjorde design, testing og implementering av en ny IP-kjerne uten inngående kunnskap om VHDL programmering.

Table of Content

List of figures

List of code

Acronyms

AC	Alternating Current
ADC	Analog Digital Controller
CPU	Central Processor Unit
DC	Direct Current
DSP	Digital Signal Processors
pu	Per unit
PWM	Pulse Width Modulation
FOC	Field-Oriented Control
SDK	System Development Kit
MMF	Magnetomotive Force
IGBT	Insulated-Gate Bipolar Transistor
FFT	Fast Fourier Transform
AAF	Anti-Aliasing Filter
BSP	Board Support Package

Notations

Symbol	Explanation
U	Voltage
I	Current
R	Resistance
L	Inductance
Ψ	Flux linkage
S	Apparent power
$\underline{\alpha}$	Alpha-axis
$\underline{\beta}$	Beta-axis
φ	Current angle
δ	Voltage angle
ε	Flux linkage angle
$\underline{U}$	Column matrix of voltages
$\underline{I}$	Column matrix of currents
$\underline{\Psi}$	Column matrix of flux linkages
$\boldsymbol{R}$	Resistance matrix
$\boldsymbol{L}$	Inductance matrix
R_s	Stator resistance
R_r	Rotor resistance
$L_{s\sigma}$	Stator leakage inductance
$L_{r\sigma}$	Rotor leakage inductance
L_{sm}	Peak value of mutual inductance between stator windings
L_{rm}	Peak value of mutual inductance between rotor windings
L_{srm}	Peak value of mutual inductance between stator and rotor winding
x_s	Stator reactance
x_r	Rotor reactance
x_σ	Leakage reactance
$x_{s\sigma}$	Stator leakage reactance
$x_{r\sigma}$	Rotor leakage reactance
x_m	Mutual reactance
x_M	Equivalent mutual reactance
T_r	Rotor rime constant
n	Rotor speed $[pu]$
f	Frequency

ω	Angular frequency
θ	Angle of rotor
θ_k	Angle of fictitious system with respect to stator
θ_r	Angle of fictitious system with respect to rotor
T_e	Electrical torque
τ_e	Electrical torque in pu
p	Pole pairs
$\boldsymbol{T}$	Transformation matrix
$\boldsymbol{J}$	Rotation matrix used in transformation

Small letters indicate pu values

Superscript	**Explanation**
S	Stator quantities in stator coordinates
R	Rotor quantities in rotor coordinates
SR	Stator and rotor quantities in their original coordinates
s	Referred to stator
r	Referred to rotor
k	Referred to transformed frame
ψ_s	Referred to stator flux linkage axis
ψ_r	Referred to rotor flux linkage axis
^	Estimated value
$*$	Actual value
$'$	Distorted value

Subscript	**Explanation**
a, b, c	Phases in the machine
α	α-axis quantity
β	β-axis quantity
s	Stator quantity
r	Rotor quantity
e	Electrical quantity
N	Rated value
n	Reference value
$-$	Vector

1 Introduction

1.1 Background and Motivation

Ever since the introduction of variable speed drives, pioneered in the late 70's, AC machines have taken over more and more of the variable speed applications previously dominated by DC machines [1]. Throughout the years, different control strategies have been developed to fulfil different needs in the market. These are typically divided into high-performance and low-performance drives. Traditionally, high-performance drives have required position measurements in the control scheme. However, during the last decades extensive research has been carried out in order to remove the dependency of position measurement, also known as sensorless control. Position sensor itself and wiring between sensor and controller represent a reliability issue, increased maintenance work, increased weight and unnecessary cost [2].

High-performance sensorless drives require estimation of rotor position. Position estimation can be achieved with the knowledge of flux linkages in the machine. This is exploited by the control strategy *field-oriented control* (FOC), where flux linkages are calculated using current and voltage measurements in the so-called *voltage model*. It is known that the open integrator used in the voltage model can cause a DC-offset in the flux linkage estimation if subjected to errors in parameter estimation or measurements. In literature, this phenomenon is referred to as *drifting* and the implications it causes has grates impact during low speed operation or slow runs through zero speed. Today it exists numerus methods, so-called observers, that tackles the challenges caused by drifting [3-5]. For several years, sensorless control has also been a field of interest at the Department of Electric Power Engineering at NTNU, where project, master and doctor theses have been written about the subject [6-9].

To avoid drifting it is important to apply accurate voltage and current measurements. Direct voltage measurements on inverter terminals are the most accurate, however high rise-times and high output voltage require special measuring equipment with adequate bandwidth and insulation [10]. Moreover, the analog signal processing may introduce offset errors. Expensive measuring hardware and analog signal processing can be avoided by estimating the inverter voltage output with help of the IGBTs control signal. However, semiconducting components in the inverter cause a non-linear voltage drop that has to be taken into account in order to achieve desired estimation accuracy.

To facilitate further research and development in fields like sensorless FOC, a R&D group at the department of Electrical Power Engineering has since the beginning of 2019 worked on establishing a common NTNU Control Platform. The complete drive setup uses a control board from Avnet and a process interface board developed by SINTEF Energy. The setup also uses release and monitoring tools provided by The Switch Marine Drives. The control platform will act as the foundation for future research and a starting point for master and doctor theses in the years to come.

The NTNU control platform consists of a CPU routine and a complete FPGA design. The CPU routine is written in C++ and developed in Xilinx's *System Development Kit* (SDK). FPGA design is constructed out of a network of IP-core using another Xilinx program called *Vivado*. Previously, IP-cores have been written using the language VHDL and development of IP-cores has required specialist knowledge. However, in today's market there exists development tools that can make this work significantly easier and allow a broader range of electrical engineers to participate in FPGA design. This thesis will use *Xilinx System Generator DSP* together with MATLAB *Simulink* in the development of a new IP-core. The IP-core will be a part of the output voltage estimator.

1.2 Scope of work

This thesis focuses on modelling and control of an induction machine drive. It also focuses on implementation of an on-line voltage estimation technique for a two-level three-phase inverter. IP-core development using *Xilinx System Generator DSP* at the department of Electrical Power Engineering has previously only been done to a small extent. This thesis present and explain the tool in detail with the intention of being an example for future development.

The scope of the work includes:

- Modelling, control and scaling of the induction machine according to the concepts and notations presented in the course Electrical Drives (TET4120).
- Modelling of a two-level three-phase inverter.
- Development of an on-line voltage estimator and an open loop V/f-controller.
- Analysis of simulation and test results with the goal of validating the estimation technique.

Limitations of the work includes:

- Tests of voltage estimator are not done on a physical drive.
- Current measurements are emulated from the CPU and not form a running machine.
- FOC is not implemented in the control routine.

There are also a number of simplifications and approximations in the induction machine model, inverter model and output voltage estimator. These are explained in the receptive chapters.

1.3 Organization of the report

The report is divided into eight main chapters: Physical modelling of the induction machine, Clark and Park transformation and pu scaling of voltage and flux linkage equations are presented in chapter 2. Open loop V/f-control and closed loop FOC alongside with the phenomenon drifting are presented in chapter 3. A non-ideal inverter model and consequences of errors in current measurements are analysed in chapter 4. Chapter 5 introduces the programming structure for the NTNU Control Platform and gives an example of how to generate an IP-core using Xilinx's System Generator tool in MATLAB Simulink. A summary of CPU routines for the drive and voltage estimator, together with a thorough explanation of the developed IP-core, is presented in chapter 6. Chapters 7 presents simulation and control board tests. A discussion of the thesis is given in chapter 8. Finally, the conclusions and suggestion for further work are given in chapter 9.

2 Modelling and Scaling of the Induction Machine

2.1 Introduction to the IM machine.

The asynchronous machine is a type of AC machine and is the most commonly used electrical machine in the industry. This is because of its simple construction, easy maintenance, robustness and versatile area of application. The output power of the asynchronous machine can range from a few watts up towards $10\,MW$ [11]. The operating principles of the machine are described by Faraday's law of induction and Lorentz force on the conductors in the rotor. Alternating currents in the stator windings set up a magnetic field in the machine's stator and rotor. When the alternating field crosses the rotor conductors a current is induced within the conductor, thus creating a force perpendicular to the field and current.

Asynchronous machines come with two different types of rotors, wounded and squirrel-cage rotor. Asynchronous machines with a wounded rotor, also called double fed asynchronous machines, have isolated windings in both stator and rotor. The winding has the same number of poles and phases as the stator and each phase is connected to its own isolated slipring. Additional resistance can be connected in series with the rotor winding during start-up to increase the starting torque. As the machine accelerates, the resistances are short circuited to increase the torque at higher speeds and to reduce conduction losses [12]. The other and more commonly used asynchronous machine is the one using a squirrel-cage rotor. This machine is commonly called *Induction Machine* (IM). The squirrel-cage rotor is made out of multiple aluminium or copper rods that are moulded in slots of the rotor and short circuited at both ends. Lack of rotor windings and slipring with brushes makes the induction machine both cheaper to produce and easier to maintain than the double fed asynchronous machine. In this master thesis, modelling and control will be based on the induction machine.

An illustration of the three-phase induction machine is presented in Figure 2.1. Each distributed phase winding is represented by an equivalent concentrated coil aligned with its respective magnetic axis. The three magnetic axes are shifted $120°$ in space and is shown as $\underline{a}^S, \underline{b}^S, \underline{c}^S$ in the figure.

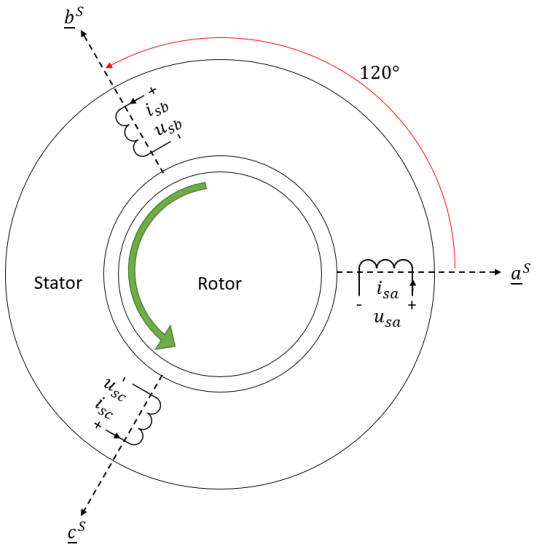


Figure 2.1: Principle drawing of the three-phase Induction Machine

2.2 Modelling of the Induction Machine

There are several different reasons why it is important to make equivalent models of the induction machine. Different control strategies use different transformed models of the physical machine in order to simplify the analysis and equations [11]. The models also allow the machine and control strategies to be tested using simulation tools.

The control and analysis used in this master thesis require two different transformed models. Stator oriented model and rotor-flux oriented model. Both models have fictitious stator and rotor windings located at the same magnetic axis. The transformed models are presented in detail in this subchapter. However, before deducting the different transformed models, it is necessary to make a physical model representing the real currents, voltages, impedances and fluxes in the machine. Moreover, to simplify the transformations it is possible to make a two-phase representation of the physical three-phase system.

For all the models presented in this thesis, direction and polarity of currents and voltages are defined according to a machine running in motor operation. Capital symbols are used for physical values and lower-case letters are used for pu-values. The superscript S and R indicate that the vector is in its original frame of reference, i.e. stator quantiles in stator frame and rotor quantiles in rotor frame. Lower case superscript indicates what reference frame the windings are fixed to, i.e. s is fixed to stator, r is fixed to rotor or k is fixed to an unspecified system. Subscript s and r indicate where the winding is located. Furthermore, subscript a, b, c or α, β specify the stator or rotor winging in question.

The physical machine model presented is based on the work done in [11] and [13] and both Clarke and Park transformations are based on the work done in [11].

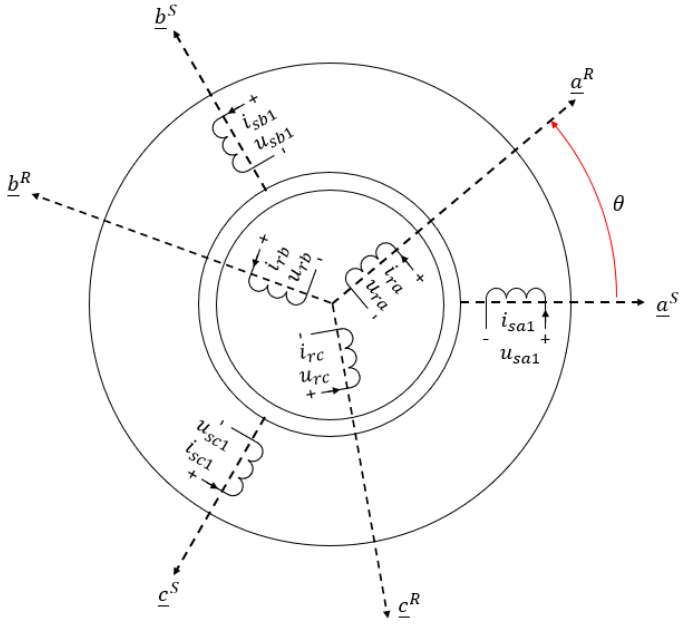


Figure 2.2: Schematic representation of the three-phase Induction Machine

The following five assumptions are taken into account when modelling the machine:

1. The stator windings only give a sinusoidal distributed field in the air gap.
2. All the stator windings are equal and have their unique orientation.
3. Resistance and inductances are assumed independent of temperature and frequency.
4. Magnetic saturation, hysteresis and eddy currents are neglected.
5. The squirrel-cage rotor is equivalent to a wounded rotor.

2.2.1 Physical model

The physical model is based on the so-called Co-energy concept and can be constructed by using the voltage balance equations for each of the equivalent stator and rotor winding [11]. Eq.(2.1) shows voltage equation for the windings in the induction machine given in matrix form. The six windings in question are presented in Figure 2.2. Flux linkage in the voltage equation can be expressed by eq.(2.2).

$$\underline{U}^{SR} = \boldsymbol{R}^{SR} \cdot \underline{I}^{SR} + \frac{d\underline{\Psi}^{SR}}{dt} \tag{2.1}$$

$$\underline{\Psi}^{SR} = \boldsymbol{L}^{SR} \cdot \underline{I}^{SR} \tag{2.2}$$

Where the voltage, current and flux linkage vectors are:

$$\begin{aligned} \underline{U}^{SR} &= [U_{sa} \quad U_{sb} \quad U_{sc} \quad U_{ra} \quad U_{rb} \quad U_{rc}]^T \\ \underline{I}^{SR} &= [I_{sa} \quad I_{sb} \quad I_{sc} \quad I_{ra} \quad I_{rb} \quad I_{rc}]^T \\ \underline{\Psi}^{SR} &= [\Psi_{sa} \quad \Psi_{sb} \quad \Psi_{sc} \quad \Psi_{ra} \quad \Psi_{rb} \quad \Psi_{rc}]^T \end{aligned} \tag{2.3}$$

The resistance matrix containing the physical stator and rotor resistance has the shape of a $\boldsymbol{I}_4$ identity matrix and can be expressed as:

$$\boldsymbol{R}^{SR} = diag[R_s \quad R_s \quad R_s \quad R_r \quad R_r \quad R_r] \tag{2.4}$$

The inductances used to calculate the flux linkages in the machine can be expressed with help of submatrices. $\boldsymbol{L}_{ss}^{S}$ and $\boldsymbol{L}_{rr}^{S}$ are the self and mutual inductances of stator and rotor with respect to itself. The air gap between stator and rotor is assumed to be constant since the rotor is assumed to be symmetrical, thus making both submatrices independent of rotor angle, θ. $\boldsymbol{L}_{sr}^{S}(\theta)$ and $\boldsymbol{L}_{rs}^{R}(\theta)$ are the mutual inductance between stator and rotor and vice versa, both with a rotor angle dependency.

$$\boldsymbol{L}^{SR} = \begin{bmatrix} \boldsymbol{L}_{ss}^{S} & \boldsymbol{L}_{sr}^{S}(\theta) \\ \boldsymbol{L}_{rs}^{R}(\theta) & \boldsymbol{L}_{rr}^{S} \end{bmatrix} \tag{2.5}$$

Submatrices $\boldsymbol{L}_{ss}^{S}$ are written in detail in eq.(2.6). The diagonal elements show the self-inductance of each winding. Self-inductance is given by stator leakage induces, $L_{s\sigma}$ (i.e. inductance in the end windings etc.) and mutual inductance of the winding with respect to itself (i.e. $L_{sm} \cdot \cos(0)$). The off-diagonal elements give the mutual inductance between two different windings. From Figure 2.2 it can be seen that the stator windings are located $120°$ from each other, thus making the off-diagonal elements equal $L_{sm} \cdot \pm\cos(2\pi/3)$.

$$\boldsymbol{L}_{ss}^{S} = \begin{bmatrix} L_{sa} & L_{sasb} & L_{sasc} \\ L_{sbsa} & L_{sb} & L_{sbsc} \\ L_{scsa} & L_{scsb} & L_{sc} \end{bmatrix} = L_{s\sigma} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + L_{sm} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & -\frac{1}{2} & 1 \end{bmatrix} \tag{2.6}$$

Same goes for self and mutual inductances in the equivalent rotor windings, $\boldsymbol{L}_{rr}^{R}$.

$$\boldsymbol{L}_{rr}^{R} = \begin{bmatrix} L_{ra} & L_{rarb} & L_{rarc} \\ L_{rbra} & L_{rb} & L_{rbrc} \\ L_{rcra} & L_{rcrb} & L_{rc} \end{bmatrix} = L_{r\sigma} \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} + L_{rm} \begin{bmatrix} 1 & -\frac{1}{2} & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & -\frac{1}{2} & 1 \end{bmatrix} \tag{2.7}$$

The mutual inductance between stator and rotor given by $\boldsymbol{L}_{sr}^{S}$ is presented in the matrix below. The mutual inductances are dependent on rotor position and is given by the peak inductance between a stator and rotor winding, L_{srm}, and the physical angle between the two windings at a given time in space.

$$\boldsymbol{L}_{sr}^{S}(\theta) = L_{srm}\begin{bmatrix} \cos(\theta) & \cos\left(\theta+\frac{2\pi}{3}\right) & \cos\left(\theta+\frac{4\pi}{3}\right) \\ \cos\left(\theta-\frac{2\pi}{3}\right) & \cos(\theta) & \cos\left(\theta+\frac{2\pi}{3}\right) \\ \cos\left(\theta-\frac{4\pi}{3}\right) & \cos\left(\theta-\frac{2\pi}{3}\right) & \cos(\theta) \end{bmatrix} \tag{2.8}$$

The mutual inductance between rotor and stator, $\boldsymbol{L}_{rs}^{R}(\theta)$, is simply the transposed matrix of $\boldsymbol{L}_{sr}^{S}(\theta)$.

$$\boldsymbol{L}_{rs}^{R}(\theta) = \left(\boldsymbol{L}_{sr}^{S}(\theta)\right)^{T} \tag{2.9}$$

The different constants used in the inductance matrices are summarised below

$L_{s\sigma}$ – Leakage inductance of each stator winding
$L_{r\sigma}$ – Leakage inductance of each rotor winding
L_{sm} – Peak value of mutual inductance between two stator windings
L_{rm} – Peak value of mutual inductance between two rotor windings
L_{srm} – Peak value of mutual inductance between a stator and rotor winding

2.2.2 Clarke Transformation

The sinusoidal phase currents, voltages and flux linkages in the machine can be represented as three vectors, shifted 120° in space, in the two-dimensional plane. The sum of the phase vectors will be a rotating vector with constant amplitude. This rotating space vector can also be represented by a two-phase winding where the windings are shifted by 90 electrical degrees. The transformation from a three-phase system to a two-phase system is usually called Clark transformation, after professor Edith Clark, or α, β, γ transformation. Preforming a Clark transformation will simplify both equations and analysis of the system.

The old three-phase system and the transformed two-phase system are presented in Figure 2.3 and Figure 2.4, respectively. It can be seen that the new α-axis is located similar to the old a-axis. β-axis is leading α-axis with 90 electrical degrees. The angle θ are the same in both models.

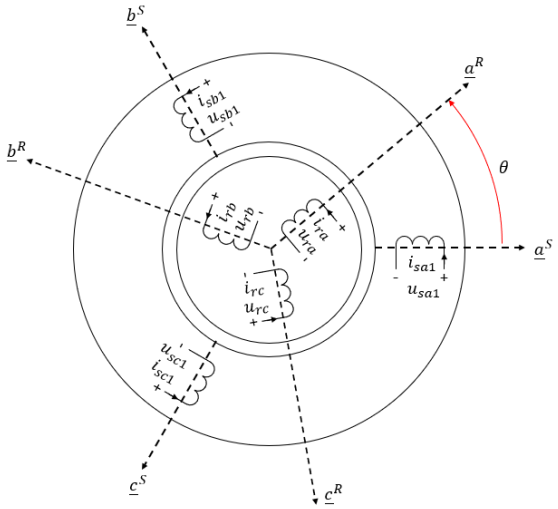


Figure 2.3: Three-phase induction machine

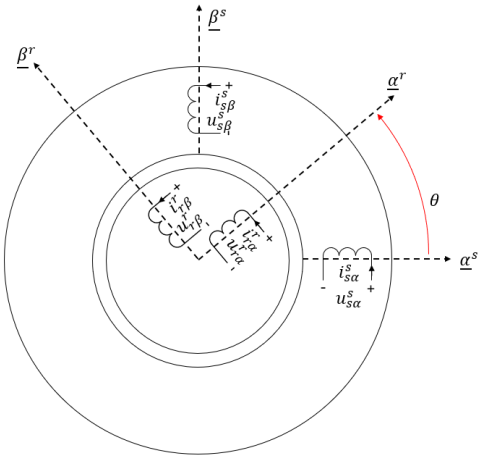


Figure 2.4: Equivalent two-phase induction machine

Two-phase transformation of stator and rotor current can be done using the equations presented in (2.10). The two-phase representation of the three-phase system as shown in Figure 2.4 and given by the equations are only valid if it is assumed that the sum of current in the three-phase system is equal to zero. However, this will not limit the model since the drive will be tripped if current starts returning through earth, thus making the sum of phase current different from zero.

Stator currents	*Rotor currents*
$I_{s\alpha}^{s} = \frac{2}{3}\left(I_{sa} - \frac{1}{2}\cdot I_{sb} - \frac{1}{2}I_{sc}\right) = I_{sa}$	$I_{r\alpha}^{r} = \frac{2}{3}\left(I_{ra} - \frac{1}{2}\cdot I_{rb} - \frac{1}{2}I_{rc}\right) = I_{ra}$
$I_{s\beta}^{s} = \frac{2}{3}\left(\frac{\sqrt{3}}{2}\cdot I_{sb} - \frac{\sqrt{3}}{2}I_{sc}\right) = \frac{\sqrt{3}}{3}(I_{sb} - I_{sc})$	$I_{r\beta}^{r} = \frac{2}{3}\left(\frac{\sqrt{3}}{2}\cdot I_{rb} - \frac{\sqrt{3}}{2}I_{rc}\right) = \frac{\sqrt{3}}{3}(I_{rb} - I_{rc})$

(2.10)

Equations needed to transform currents in the two-phase system back to a three-phase system are presented in (2.11).

Stator currents	*Rotor currents*
$I_{sa} = I_{s\alpha}^{s}$	$I_{ra} = I_{r\alpha}^{r}$
$I_{sb} = -\frac{1}{2}\cdot I_{s\alpha}^{s} + \frac{\sqrt{3}}{2}\cdot I_{s\beta}^{s}$	$I_{rb} = -\frac{1}{2}\cdot I_{r\alpha}^{r} + \frac{\sqrt{3}}{2}\cdot I_{r\beta}^{r}$
$I_{sc} = -\frac{1}{2}\cdot I_{s\alpha}^{s} - \frac{\sqrt{3}}{2}\cdot I_{s\beta}^{s}$	$I_{rc} = -\frac{1}{2}\cdot I_{r\alpha}^{r} - \frac{\sqrt{3}}{2}\cdot I_{r\beta}^{r}$

(2.11)

Voltages and flux linkages can be transformed from a three-phase to a two-phase representation and vice versa using the same relations as shown in (2.10) and (2.11). As for currents, the transformed voltage and flux linkage model is only valid if the sum of phase voltages is zero.

After the Clark transformation, voltage and flux linkage equations can be expressed in matrix form by eq.(2.12) and (2.13), respectively.

$$\underline{U}^{sr} = \boldsymbol{R}^{sr}\cdot\underline{I}^{sr} + \frac{d\underline{\Psi}^{sr}}{dt} \tag{2.12}$$

$$\underline{\Psi}^{sr} = \boldsymbol{L}^{sr}\cdot\underline{I}^{sr} \tag{2.13}$$

Where the voltage, current and flux linkage vectors are:

$$\begin{aligned}\underline{U}^{sr} &= \begin{bmatrix}U_{s\alpha}^{s} & U_{s\beta}^{s} & U_{r\alpha}^{r} & U_{r\beta}^{r}\end{bmatrix}^{T}\\ \underline{I}^{sr} &= \begin{bmatrix}I_{s\alpha}^{s} & I_{s\beta}^{s} & I_{r\alpha}^{r} & I_{r\beta}^{r}\end{bmatrix}^{T}\\ \underline{\Psi}^{sr} &= \begin{bmatrix}\Psi_{s\alpha}^{s} & \Psi_{s\beta}^{s} & \Psi_{r\alpha}^{r} & \Psi_{r\beta}^{r}\end{bmatrix}^{T}\end{aligned} \tag{2.14}$$

The resistance matrix is given by the diagonal matrix shown in (2.15). The components of the matrix are the physical resistance in stator and rotor windings.

$$\boldsymbol{R}^{sr} = diag[R_s \quad R_s \quad R_r \quad R_r] \tag{2.15}$$

In order to use the same inductances as in the physical model, a scaling factor of $3/2$ must be introduced to the elements that produces flux in the air gap. This is done to compensate for the MMF produced by the two-phase system that would otherwise be $2/3$ smaller than the real MMF [11].

The new inductance matrix for the two-phase system is presented in (2.16). Comparing $\boldsymbol{L}^{sr}$ with $\boldsymbol{L}^{SR}$ from the physical model, it can be seen that the matrix is reduced from a $6x6$ to a $4x4$ matrix. Moreover, since the new stator and rotor windings are shifted by 90°, mutual inductance between the two windings in stator or rotor is zero. Mutual inductance between a stator and rotor winding is still dependent on rotor position.

$$\boldsymbol{L}^{sr}(\theta) = \begin{bmatrix} L_{s\alpha\alpha} & 0 & L_{s\alpha r\alpha}(\theta) & L_{s\alpha r\beta}(\theta) \\ 0 & L_{s\beta\beta} & L_{s\beta r\alpha}(\theta) & L_{s\beta r\beta}(\theta) \\ L_{r\alpha s\alpha}(\theta) & L_{r\alpha s\beta}(\theta) & L_{r\alpha\alpha} & 0 \\ L_{r\beta s\alpha}(\theta) & L_{r\beta s\beta}(\theta) & 0 & L_{s\beta\beta} \end{bmatrix}$$

$$= \begin{bmatrix} L_{s\sigma} + \frac{3}{2}L_{sm} & 0 & \frac{3}{2}L_{srm}\cos\theta & -\frac{3}{2}L_{srm}\sin\theta \\ 0 & L_{s\sigma} + \frac{3}{2}L_{sm} & \frac{3}{2}L_{srm}\sin\theta & \frac{3}{2}L_{srm}\cos\theta \\ \frac{3}{2}L_{srm}\cos\theta & \frac{3}{2}L_{srm}\sin\theta & L_{r\sigma} + \frac{3}{2}L_{rm} & 0 \\ -\frac{3}{2}L_{srm}\sin\theta & \frac{3}{2}L_{srm}\cos\theta & 0 & L_{r\sigma} + \frac{3}{2}L_{rm} \end{bmatrix} \quad (2.16)$$

Where: $L_{s\sigma}$ – Leakage inductance of each stator winding
$L_{r\sigma}$ – Leakage inductance of each rotor winding
L_{sm} – Peak value of mutual inductance between two stator windings
L_{rm} – Peak value of mutual inductance between two rotor windings
L_{srm} – Peak value of mutual inductance between a stator and rotor winding

2.2.3 Park Transformation

From last section, it was shown that the inductance matrix, $\boldsymbol{L}^{sr}$, was dependent on rotor position. This dependency occurs because stator and rotor windings are fixed to different axis in the model. By modelling both stator and rotor winding in the same axis system, inductance becomes independent of position. This will simplify the model even further.

The transformation from the two-phase equivalent system to the new system with stator and rotor windings rotating in the same axis system is called Park transformation. The Park transformed system is show in Figure 2.6. The figure shows the new fictitious stator and rotor winding fixed to the same axis.

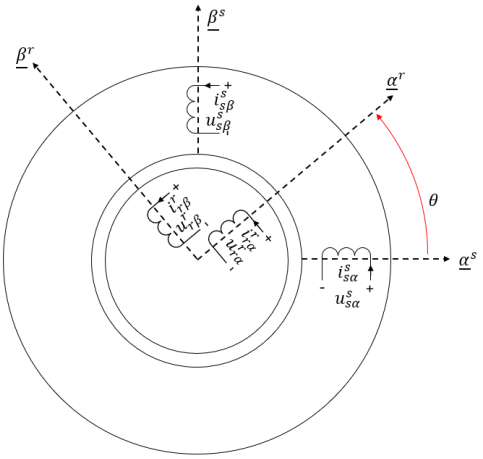


Figure 2.5: Clarke transformed model

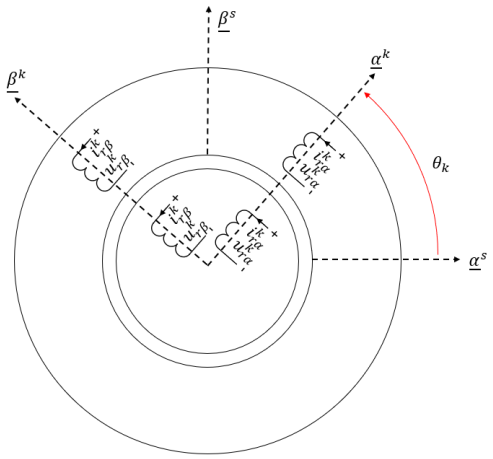


Figure 2.6:Park transformed model

The transformation matrix and the inverse transformation matrix used in Park transformation are presented in (2.17).

$$\boldsymbol{T}^{k} = \begin{bmatrix} \boldsymbol{T}_{ss}^{k} & 0 \\ 0 & \boldsymbol{T}_{rr}^{k} \end{bmatrix} \qquad \boldsymbol{T}^{-k} = \begin{bmatrix} \boldsymbol{T}_{ss}^{-k} & 0 \\ 0 & \boldsymbol{T}_{rr}^{-k} \end{bmatrix} \tag{2.17}$$

Where the different elements of the transformation matrices are:

$$\boldsymbol{T}_{ss}^{k} = \begin{bmatrix} \cos\theta_k & \sin\theta_k \\ -\sin\theta_k & \cos\theta_k \end{bmatrix} \qquad \boldsymbol{T}_{ss}^{-k} = \begin{bmatrix} \cos\theta_k & -\sin\theta_k \\ \sin\theta_k & \cos\theta_k \end{bmatrix} \tag{2.18}$$

$$\boldsymbol{T}_{rr}^{k} = \begin{bmatrix} \cos\theta_r & \sin\theta_r \\ -\sin\theta_r & \cos\theta_r \end{bmatrix} \qquad \boldsymbol{T}_{rr}^{-k} = \begin{bmatrix} \cos\theta_r & -\sin\theta_r \\ \sin\theta_r & \cos\theta_r \end{bmatrix} \tag{2.19}$$

Where: $\theta_r = \theta_k - \theta$

The relation between voltages, currents and impedances between Park and Clarke transformed systems are given by:

$$\begin{aligned} \underline{U}^{k} &= \boldsymbol{T}^{k} \cdot \underline{U}^{sr} & \qquad \underline{U}^{sr} &= \boldsymbol{T}^{-k} \cdot \underline{U}^{k} \\ \underline{I}^{k} &= \boldsymbol{T}^{k} \cdot \underline{I}^{sr} & \qquad \underline{I}^{sr} &= \boldsymbol{T}^{-k} \cdot \underline{I}^{k} \\ \boldsymbol{X}^{k} &= \boldsymbol{T}^{k} \cdot \boldsymbol{X}^{sr} \cdot \boldsymbol{T}^{-k} \end{aligned} \tag{2.20}$$

An expression for the flux linkage can be found by multiplying both sides of eq. (2.13) with $\boldsymbol{T}^{k}$ and using the relations given in (2.20).

$$\underline{\Psi}^{\mathrm{k}} = \boldsymbol{T}^{k} \cdot \underline{\Psi}^{\mathrm{sr}} = \boldsymbol{T}^{k} \cdot \boldsymbol{L}^{sr} \cdot \boldsymbol{T}^{-k} \cdot \underline{I}^{k} = \boldsymbol{L}^{k} \cdot \underline{I}^{k} \tag{2.21}$$

Voltage equations in the new transformed system can be found by multiplying both sides of the voltage equation, eq.(2.12), with the transformation matrix, $\boldsymbol{T}^{k}$, and using the relations between the two systems.

$$\begin{aligned} \boldsymbol{T}^{k} \cdot \underline{U}^{sr} &= \boldsymbol{T}^{k}\left(\boldsymbol{R}^{sr} \cdot \underline{I}^{sr} + \frac{d\underline{\Psi}^{sr}}{dt}\right) \\ \underline{U}^{k} &= \boldsymbol{T}^{k} \cdot \boldsymbol{R}^{sr} \cdot \boldsymbol{T}^{-k} \cdot \underline{I}^{k} + \boldsymbol{T}^{k} \cdot \frac{d\left(\boldsymbol{T}^{-k} \cdot \underline{\Psi}^{k}\right)}{dt} \\ \underline{U}^{k} &= \boldsymbol{R}^{k} \cdot \underline{I}^{k} + \frac{d\underline{\Psi}^{k}}{dt} + \boldsymbol{T}^{k} \cdot \frac{d\boldsymbol{T}^{-k}}{dt} \cdot \underline{\Psi}^{k} \end{aligned} \tag{2.22}$$

It can be seen that the inverse transformation matrix, $\boldsymbol{T}^{-k}$, is a function of θ_r and θ_k,. The derivative of $\boldsymbol{T}^{-k}$ can be expressed as:

$$\frac{d\boldsymbol{T}_{(\theta_r,\theta_k)}^{-k}}{dt} \cdot \frac{(\partial\theta_r \cdot \partial\theta_k)}{(\partial\theta_r \cdot \partial\theta_k)} = \frac{\partial\theta_r}{dt} \cdot \frac{\partial\boldsymbol{T}^{-k}}{\partial\theta_r} + \frac{\partial\theta_k}{dt} \cdot \frac{\partial\boldsymbol{T}^{-k}}{\partial\theta_k} = \omega_n \cdot f_r \cdot \frac{\partial\boldsymbol{T}^{-k}}{\partial\theta_r} + \omega_n \cdot f_k \cdot \frac{\partial\boldsymbol{T}^{-k}}{\partial\theta_k} \tag{2.23}$$

By substituting eq.(2.23) in eq.(2.22) and introducing the coupling matrices $\boldsymbol{J}_r$ and $\boldsymbol{J}_k$, the voltage equation can be expressed as:

$$\underline{U}^{k} = \boldsymbol{R}^{k} \cdot \underline{I}^{k} + \frac{d\underline{\Psi}^{k}}{dt} + \omega_n \cdot f_r \cdot \boldsymbol{J}_r \cdot \underline{\Psi}^{k} + \omega_n \cdot f_k \cdot \boldsymbol{J}_k \cdot \underline{\Psi}^{k} \tag{2.24}$$

Where: $\boldsymbol{J}_r = T^{-k} \cdot \frac{dT^{-k}}{d\theta_r} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$ $\qquad \boldsymbol{J}_k = T^{-k} \cdot \frac{dT^{-k}}{d\theta_k} = \begin{bmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

Voltage, current and flux linkage vectors in the new system become:

$$\begin{aligned}\underline{U}^k &= \begin{bmatrix} U_{s\alpha}^k & U_{s\beta}^k & U_{r\alpha}^k & U_{r\beta}^k \end{bmatrix}^T \\ \underline{I}^k &= \begin{bmatrix} I_{s\alpha}^k & I_{s\beta}^k & I_{r\alpha}^k & I_{r\beta}^k \end{bmatrix}^T \\ \underline{\Psi}^k &= \begin{bmatrix} \Psi_{s\alpha}^k & \Psi_{s\beta}^k & \Psi_{r\alpha}^k & \Psi_{r\beta}^k \end{bmatrix}^T \end{aligned} \tag{2.25}$$

The resistance matrix becomes equal to that used in previous section.

$$\boldsymbol{R}^k = diag[R_s \quad R_s \quad R_r \quad R_r] \tag{2.26}$$

As mention earlier, one of the main motivations of preforming a Park transformation is the inductance matrix's independency of rotor position. Since stator and rotor winding are fixed to the same axis system, the new inductance matrix simply becomes $\boldsymbol{L}^{sr}(\theta = 0)$.

$$\boldsymbol{L}^k = \boldsymbol{L}^{sr}(0) = \begin{bmatrix} L_{s\sigma} + \frac{3}{2}L_m & 0 & \frac{3}{2}L_m & 0 \\ 0 & L_{s\sigma} + \frac{3}{2}L_m & 0 & \frac{3}{2}L_m \\ \frac{3}{2}L_m & 0 & L_{r\sigma} + \frac{3}{2}L_m & 0 \\ 0 & \frac{3}{2}L_m & 0 & L_{r\sigma} + \frac{3}{2}L_m \end{bmatrix} \tag{2.27}$$

Where: $L_{s\sigma}$ – Leakage inductance of each stator winding
$L_{r\sigma}$ – Leakage inductance of each rotor winding
L_m – Mutual inductance between two windings

Orientation of axis system

For a synchronous machine with salient poles, Park transformed system is always oriented to the rotor in order to achieve an inductance matrix independent of the position. In an asynchronous machine however, axis orientation does not have the same constraints. The symmetrical rotor ensures a position independent inductance matrix as long as the equivalent stator and rotor windings are orientated to the same magnetic axis.

There are several different alternatives on how to orient the Park transformed system. The desired orientation is dependent on application, control and measurement available of the machine. Commonly used orientations include: Stator-oriented model, rotor-oriented model and space-vector oriented model. In this thesis, stator-oriented model and rotor-flux (space-vector) oriented model will be used.

Stator-oriented model

In stator-oriented model, $\underline{\alpha}^k, \underline{\beta}^k$ is fixed to the $\underline{\alpha}^s, \underline{\beta}^s$-axis, thus making the new axis system stationary. Currents and voltages are alternating with stator frequency, f_s, in both equivalent stator and rotor windings. This can for example be beneficial when using a hysteresis controller.

$$\theta_k = 0 \qquad \theta_r = -\theta \qquad f_k = 0 \tag{2.28}$$

Rotor-flux oriented model

In rotor-flux oriented system, $\underline{\alpha}^k$ is fixes to the rotor flux $\underline{\psi_r}$. The rotor flux rotates with the same frequency as the stator frequency i.e. synchronous frequency. The synchronous rotating model result in DC-quantities of currents and voltages in steady state operation. This can be beneficial when using PI-controllers.

$$\theta_k = \theta_{\psi_r} \qquad \theta_r = \theta_{\psi_r} - \theta \qquad f_k = f_{\psi_r} = f_s \tag{2.29}$$

Electrical torque

The electrical torque of the machine in the Park transformed model can be expressed as [11]:

$$T_e = \frac{p}{2} \cdot 3 \cdot L_m \cdot (I_{s\beta}^k \cdot I_{r\alpha}^k - I_{s\alpha}^k \cdot I_{r\beta}^k) \tag{2.30}$$

By realizing that $\Psi_s = L_m \cdot I_r$, the torque equation can be expressed in terms of stator quantities

$$T_e = \frac{p}{2} \cdot 3 \cdot (\Psi_{s\alpha}^k \cdot I_{s\beta}^k - \Psi_{s\alpha}^k \cdot I_{s\alpha}^k) \tag{2.31}$$

Where: p is the number of pole pairs in the motor

2.3 Pu system

Once the proper transformation model has been established, it is possible to scale the values according to a pu-system. The base values are often chosen to be related to the rated value of the machine. There are many reasons why a pu-scaled model can be beneficial. Since the values are scaled according to the rated limits, it becomes easy to detect if the machine is running in overload regardless of the power rating. The scaling also makes it easier to utilize experiences with machines of other power rating. Moreover, proper pu-scaling can also simplify machine model, thus simplifying the equations [11].

Base values for stator quantities in the pu-model are shown in (2.32).

$$\begin{aligned} U_{s,basis} &= \hat{\mathrm{U}}_n & I_{s,basis} &= \hat{\mathrm{I}}_n \\ Z_{s,basis} &= \frac{\hat{\mathrm{U}}_n}{\hat{\mathrm{I}}_n} & S_{s,basis} &= \frac{3}{2} \cdot \hat{\mathrm{U}}_n \cdot \hat{\mathrm{I}}_n \\ L_{s,basis} &= \frac{\Psi_{s,basis}}{\hat{I}_n} & \Psi_{s,basis} &= \frac{\hat{\mathrm{U}}_n}{2\pi \cdot f_n} \end{aligned} \tag{2.32}$$

An important simplification that can be achieved when scaling the model is the simplified inductance matrix. By properly selecting the rotor base values, inductance matrix becomes [14]:

$$\boldsymbol{x}^k = \boldsymbol{\Psi}_{sr,base}^{-1} \cdot \boldsymbol{X}_{sr}^k \cdot \hat{\boldsymbol{I}}_{sr,basis} = \begin{bmatrix} x_s & 0 & x_m & 0 \\ 0 & x_s & 0 & x_m \\ x_m & 0 & x_r & 0 \\ 0 & x_m & 0 & x_r \end{bmatrix} \tag{2.33}$$

Where $x_s = x_m + x_{s\sigma}$
$x_r = x_m + x_{r\sigma}$

The base value for the torque is given as:

$$T_{basis} = \frac{S_n}{\Omega_n} = \frac{3}{2} \cdot p \cdot \frac{\hat{\mathrm{U}}_n \cdot \hat{\mathrm{I}}_n}{\omega_n} = \frac{3}{2} \cdot p \cdot \Psi_{\mathrm{n}} \cdot \hat{\mathrm{I}}_n \tag{2.34}$$

By scaling the voltage equations, (2.24), and flux linkage equations (2.21) according to (2.32) and introducing the scaled inductance matrix, pu-equation for voltage and flux linkage can be expressed by (2.35) and (2.36), respectively.

Voltage equations

$$\begin{aligned} u_{s\alpha}^k &= r_s \cdot i_{s\alpha}^k + \frac{1}{\omega_n} \cdot \frac{d\psi_{s\alpha}^k}{dt} - f_k \cdot \psi_{s\beta}^{\mathrm{k}} & u_{r\alpha}^k &= r_r \cdot i_{r\alpha}^k + \frac{1}{\omega_n} \cdot \frac{d\psi_{r\alpha}^k}{dt} - (f_k - n) \cdot \psi_{r\beta}^{\mathrm{k}} \\ u_{s\beta}^k &= r_s \cdot i_{s\beta}^k + \frac{1}{\omega_n} \cdot \frac{d\psi_{s\beta}^k}{dt} + f_k \cdot \psi_{s\alpha}^{\mathrm{k}} & u_{r\beta}^k &= r_r \cdot i_{r\beta}^k + \frac{1}{\omega_n} \cdot \frac{d\psi_{r\beta}^k}{dt} - (f_k - n) \cdot \psi_{r\alpha}^{\mathrm{k}} \end{aligned} \tag{2.35}$$

Flux linkage equations

$$\begin{aligned} \psi_{s\alpha}^k &= x_s \cdot i_{s\alpha}^k + x_m \cdot i_{r\alpha}^{\mathrm{k}} & \psi_{r\alpha}^k &= x_r \cdot i_{r\alpha}^k + x_m \cdot i_{s\alpha}^k \\ \psi_{s\beta}^k &= x_s \cdot i_{s\beta}^k + x_m \cdot i_{r\beta}^{\mathrm{k}} & \psi_{r\beta}^k &= x_r \cdot i_{r\beta}^k + x_m \cdot i_{s\beta}^k \end{aligned} \tag{2.36}$$

The electrical torque in pu can be expressed as follows:

$$\tau_e = \frac{T_e}{T_{basis}} = \frac{3/2 \cdot p \cdot \left(\Psi_{s\alpha}^k \cdot I_{s\beta}^k - \Psi_{s\beta}^k \cdot I_{s\alpha}^k\right)}{3/2 \cdot p \cdot \Psi_n \cdot \hat{I}_n} = \psi_{s\alpha}^k \cdot i_{s\beta}^k - \psi_{s\beta}^k \cdot i_{s\alpha}^k \tag{2.37}$$

3 Control of the Induction Machine

3.1 V/f-Control

Scalar control or more commonly known V/f-control, is the most used control strategy in the industry when controlling an induction motor. V/f-control offers a low-performance control of the machine, i.e. ability to control steady state speed but not dynamic processes between two steps in speed reference. This behaviour makes V/f-control suitable for applications with little requirement with regard to dynamic processes like fans and pumps [15].

V/f-control can be performed using both open and closed loop. The difference between the two setups is usage of position sensor. Closed loop control is able to adjust the mechanical speed to the reference value, while the open loop control regulates the synchronous rotating field according to a reference, thus giving a small steady state deviation between reference and actual speed. Both control methods use current measurements to protect the machine for overload.

The reason why the control strategy is called V/f-control is simply because the relation between stator voltage and frequency is kept constant during different operation speeds. Equations for steady state stator voltage and stator flux for low-, medium- and high-speed operation are presented in (3.1). Looking at stator flux during medium- and high-speed operation, it is evident that stator flux is proportional with stator voltage and inverse proportional with stator frequency. Consequently, by keeping the V/f ratio constant, stator flux is also kept constant. This constant relationship is shown in Figure 3.1.

$$\begin{aligned} &\textit{Low speed} && \textit{Medium and high speed} \\ &\underline{u}_s = r_s \cdot \underline{i}_s + j\left(f_s \cdot \underline{\psi}_s\right) && \underline{u}_s \approx j\left(f_s \cdot \underline{\psi}_s\right) \\ &j\underline{\psi}_s = \frac{1}{f_s} \cdot (\underline{u}_s - r_s \cdot \underline{i}_s) && j\underline{\psi}_s \approx \frac{\underline{u}_s}{f_s} \end{aligned} \tag{3.1}$$

A classical torque speed curve of an induction motor is presented in Figure 3.2. If the stator flux is kept at its rated value, hence constant V/f ratio, the familiar torque characteristic will be shifted along the frequency axis with more or less the same shape. This behaviour enables the machine to deliver nominal torque in large area of the nominal speed range. The slip rate of the machine is also kept low, giving good efficiency and low reactive power consumption during lower speed operation [16].

However, looking at voltage and flux equations during low speed operation, it is seen that resistive voltage drop starts to make an impact on the flux amplitude. The reduced flux amplitude result in a lower torque. This effect is also illustrated in Figure 3.2, where the torque at low speeds becomes lower than the nominal torque. If the machine has a high load-torque during start up, this behaviour may prove to be problematic [15].

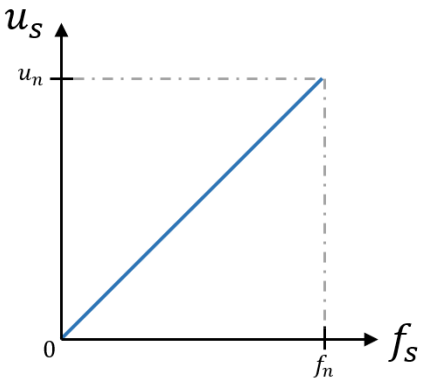


Figure 3.1: V/f curve without voltage boost

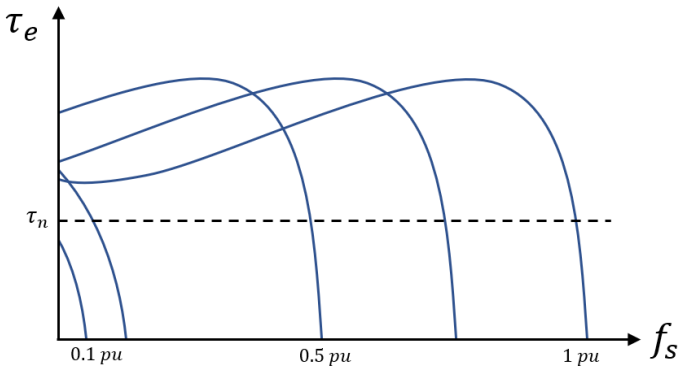


Figure 3.2: Torque/speed curve without voltage boost

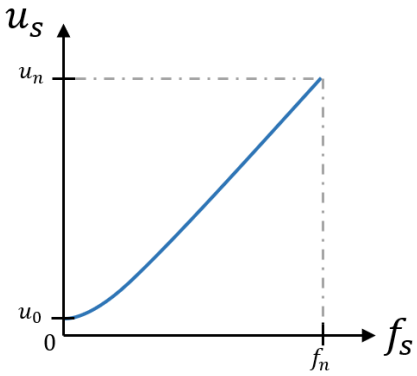


Figure 3.3: V/f curve with voltage boost

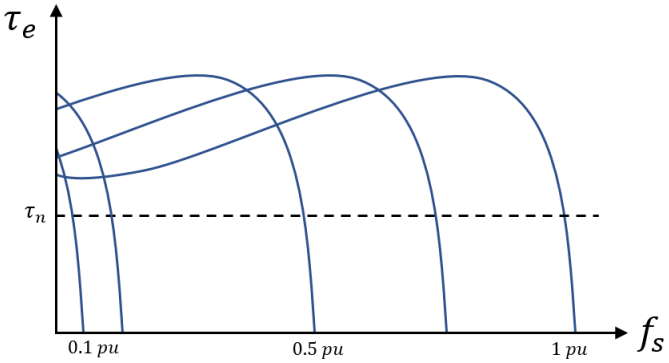


Figure 3.4: Torque/speed curve with voltage boost

One method of avoiding the problem at lower speed operation and especially when the stator frequency is close to zero is to introduce a so-called voltage boost. The voltage boost compensates for the resistive voltage drop, ensuring a stator flux at 1 pu, thus making the machine able to produce the same torque throughout the nominal speed operation range. The voltage boost and the new torque/speed characteristic are presented in Figure 3.3 and Figure 3.4, respectively [16].

3.2 Rotor-Flux Orientated Control

Some applications of the induction machines, like ship propulsion or electrical cars, may require the ability to control dynamic processes as well as steady state values of speed and torque. Such drives are called high-performance drives. A control strategy that falls under this is the so-called *field-oriented control* (FOC). FOC allows the magnitude, frequency and phase shift of the different phase-voltages in the machine to be controlled individually, thus allowing the torque and flux to change separately [13]. There are at least three different control strategies of field oriented control [11]:

1. Stator oriented field control
2. Rotor oriented field control
3. Rotor-flux oriented field control

This thesis will only present and analyse benefits and challenges regarding the rotor-flux oriented control

The rotor-flux oriented control is an attractive control strategy because the control becomes very similar to the simple DC-machine control. Since the rotor-flux oriented system is rotating with stator frequency, all steady state values of currents, voltages and fluxes become DC-quantities in steady state. Furthermore, by definition, α^k-axis is fixed to the rotor flux vector, thus $\underline{\psi}_r = \psi_{r\alpha}$ and $\psi_{r\beta} \equiv 0$. Looking at the scaled rotor-flux eq. (2.36) it becomes evident that the length of the flux is determined by $i_{s\alpha}^{\psi_r}$.

$$\psi_R = \psi_{R\alpha} \qquad \psi_{R\beta} \equiv \frac{d\psi_{R\beta}}{dt} \equiv 0 \tag{3.2}$$

Flux linkage vectors are not a physical quantity and cannot be measured. Therefore, in order to control the machine according to a flux vector it is necessary to calculate/estimate the flux vector using a flux model. Different models use different input parameters, thus making them suitable for different applications. The two estimation models presented in this thesis is called *Voltage model* and *Current model*. Usually, current model is the preferred estimation model if there is a position sender mounted on the shaft of the machine and voltage model can be used if the machine is running in sensorless control.

3.2.1 Current Model

Current model uses measurements of stator current and rotor position as input parameters. Precise position measurement and the knowledge of stator frequency allows for a Park transformed system that is fixed to the rotor flux space vector. As stated earlier, fixing the new stator and rotor winding to a system that rotates with the same frequency as stator frequency yields DC-quantities of voltage, current and flux vectors in steady state. It will be shown in the derived current model below that control of the decomposed induction machine becomes very similar to that of a DC-machine. The model presented is based on the work done in [11].

Current model is derived from the rotor voltage and rotor-flux equations presented in eq.(2.35) and (2.36). Rotor voltage vector and rotor flux vector can be presented as follows:

$$\underline{u}_r^k = r_r \cdot \underline{i}_r^k + \frac{1}{\omega_n} \cdot \frac{d\underline{\psi}_r^k}{dt} + j \cdot f_r \cdot \underline{\psi}_r^k \tag{3.3}$$

$$\underline{\psi}_r^k = \underline{i}_r^k \cdot x_r + \underline{i}_s^k \cdot x_m \tag{3.4}$$

With help of eq.(3.4), it is possible to express the rotor voltage with stator current and roto flux as state variables. Substituting for $\underline{i}_r^k$ in eq.(3.3) and rearranging the equations with the two state variables gives:

$$\underline{u}_r^k = \frac{1}{\omega_n} \cdot \frac{d\underline{\psi}_r^k}{dt} + \left(\frac{r_r}{x_r} + j \cdot f_r\right) \cdot \underline{\psi}_r^k - \frac{x_m r_r}{x_r} \cdot \underline{i}_r^k \tag{3.5}$$

It is possible to simplify the equations by introducing the following set of parameters.

$$\underline{\psi}_R^k = \frac{\underline{\psi}_r^k}{1+\sigma_r} \qquad \sigma_r = \frac{x_r}{x_m} - 1 \qquad r_R = \left(\frac{x_m}{x_r}\right)^2 r_r = \frac{r_r}{(1+\sigma_r)^2} \qquad T_r = \frac{x_r}{\omega_n r_r} \tag{3.6}$$

Since the induction machine has a rotor that is short circuited in both ends, $\underline{u}_r^k$ will always be zero. Multiplying eq.(3.5) with $1/(1+\sigma_r)$ and using the new parameters presented in (3.6), a simplified voltage equation can be expressed as:

$$\begin{aligned} \frac{\underline{u}_r^k}{1+\sigma_r} &= \frac{1}{\omega_n} \cdot \frac{d\underline{\psi}_r^k}{dt} \cdot \frac{1}{1+\sigma_r} + \left(\frac{r_r}{x_r} + j \cdot f_r\right) \cdot \frac{\underline{\psi}_r^k}{1+\sigma_r} - \frac{x_m r_r}{x_r} \cdot \frac{1}{1+\sigma_r} \cdot \underline{i}_r^k \\ 0 &= \frac{1}{\omega_n} \cdot \frac{d\underline{\psi}_R^k}{dt} + \left(\frac{1}{T_r \cdot \omega_n} + j \cdot f_r\right) \cdot \underline{\psi}_R^k - r_R \cdot \underline{i}_r^k \end{aligned} \tag{3.7}$$

By decomposing the equation above and aligning the fictitious system with the rotor-flux, eq.(3.7) can be expressed in terms of α and β components as follows:

$$0 = \frac{1}{\omega_n} \cdot \frac{d\psi_R^{\psi_r}}{dt} + \frac{\psi_R^{\psi_r}}{(\omega_n T_r)} - r_R \cdot i_{s\alpha}^{\psi_r} \tag{3.8}$$

$$0 = \frac{1}{\omega_n} \cdot \frac{d\psi_{R\beta}^{\psi_r}}{dt} + j f_r \cdot \psi_R^{\psi_r} - r_R \cdot i_{s\beta}^{\psi_r} \tag{3.9}$$

Since the α-axis is fixed to the rotor flux vector, hence flux in β-axis is equal to zero, it is possible to find an expression of rotor frequency by reengaging eq.(3.9). The rotor frequency is used to define rotor flux frequency in both dynamic and steady state operation.

$$f_r = \frac{r_R \cdot i_{s\beta}^{\psi_r}}{\psi_R^{\psi_r}} \qquad\qquad f_{\psi_r} = f_r + n \tag{3.10}$$

The differential equation describing the rotor flux is given by eq.(3.8). After multiplying the equation with ω_n and some rearrangements, the rotor flux can be described as follows:

$$\frac{d\psi_R}{dt} = -\frac{1}{T_r} \cdot \psi_R + \frac{x_M}{T_r} \cdot i_{s\alpha}^{\psi_r} \tag{3.11}$$

Where $r_R \cdot \omega_n = \frac{x_M}{T_r}$, $x_M = \frac{x_m}{(1+\sigma_r)}$

3.2.2 Voltage Model

The voltage model uses measurements of stator voltage and stator current as input parameters. There are several different ways to acquire the stator voltage. It is possible to measure the output voltage of the inverter directly using the correct measuring tools. It is also possible to estimate the stator voltage using the on and off times of the IGBTs and the DC-link voltage. Measurements of DC-link voltage is available from the inverter and requires less complicated measuring tools. Therefore, estimation of stator voltage offers a cheaper solution with less hardware required compared to the direct measurements. However, stator voltage estimation also introduces some challenges, especially in the slow speed operation area. Some of the challenges include unilinear voltage drops, blanking time and turn-on and turn-off time of the IGBTs. The impact of these non-ideal properties is discussed in chapter 4.1 and 4.2.

The main motivation for using the voltage model to estimate fluxes in the machine is because it does not require a position measurement, thus allowing for sensorless control. The model is derived from stator and rotor flux equations given in (2.36). However, using these equations requires knowledge of the stator flux. In the voltage model, stator flux is obtained by integrating the induced voltage with an open integrator, according to the stator voltage equation. Stator flux is found using following equation:

$$\underline{\hat{\psi}}_s = \omega_n \cdot \int_{t_0}^{t} \left(\underline{u}_s - \hat{r}_s \cdot \underline{i}_s\right) dt + \underline{\hat{\psi}}_s(t_0) \tag{3.12}$$

Where: $\underline{\hat{\psi}}_s(t_0)$ is the stator flux before magnetization and is equal to zero

It is important to note that the open integrator used to calculate $\underline{\hat{\psi}}_s$ is very sensitive to inaccurate estimates of stator resistance or measurements of current and voltage. These issues are explained and further discussed in chapter 3.3.

It is possible to derive the relationship between stator and rotor flux using the pu-scaled equations presented in (2.36). By using stator current and rotor flux as state variables, stator flux can be expressed as:

$$\begin{aligned} \underline{\psi}_s^k &= x_s \cdot \underline{i}_s^k + x_m \cdot \left(\frac{\underline{\psi}_r^k}{x_r} - \frac{x_m}{x_r} \cdot \underline{i}_s^k\right) \\ &= \frac{x_m}{x_r} \cdot \underline{\psi}_r^k + \left(1 - \frac{x_m^2}{x_s \cdot x_r}\right) \cdot x_s \cdot \underline{i}_s^k \end{aligned} \tag{3.13}$$

Again, the equation can be simplified by introducing some new parameters. The simplification uses the following relations:

$$\underline{\psi}_R^k = \frac{\underline{\psi}_r^k}{1+\sigma_r} \qquad x_r = (1+\sigma_r) \cdot x_m \qquad \sigma = 1 - \frac{x_m^2}{x_s \cdot x_r} \qquad x_\sigma = \sigma \cdot x_s \tag{3.14}$$

With help of the new parameters given in (3.14), eq.(3.13) simply becomes:

$$\underline{\psi}_s^k = \underline{\psi}_R^k + x_\sigma \cdot \underline{i}_s^k \tag{3.15}$$

Consequently, if stator flux is known, rotor flux becomes:

$$\underline{\psi}_R^k = \underline{\psi}_s^k - x_\sigma \cdot \underline{i}_s^k \tag{3.16}$$

Current and flux space vectors used in the voltage model are presented in Figure 3.5. The figure also shows the corresponding angles between different vectors and from vectors to the physical magnetic a-phase. In a three-phase induction machine, it is possible to use the vector calculations given by eq.(3.16) directly in order to find the rotor-flux length and position. If these calculations are to be performed, it would be natural to choose a stator-oriented Park transformed system, i.e. superscript $k = s$.

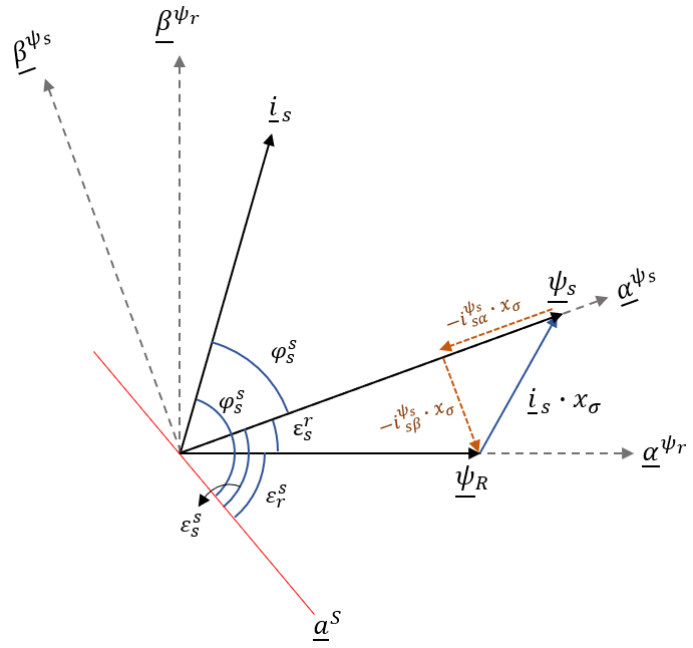


Figure 3.5: Vector diagram for used in voltage model

A more generic way of acquiring the rotor flux vector is to temporary orient the system to the stator-flux vector. It is necessary to use this approach to estimate the flux in a permanent magnet machine, synchronous machine or a multi-phase asynchronous machine. Similar with rotor-flux oriented system used in the current model, the stator flux vector is located in the $\underline{\alpha}^{\psi_s}$-axis, as shown in Figure 3.5. Angle and amplitude of stator fluxes and consequently orientation of the Park transformed system is know from integration of the flux producing voltage, i.e. eq.(3.12). Since the angles between both stator flux and current vector with respect to $\underline{a}^s$-axis is known, the angle between them simply becomes:

$$\hat{\varphi}_s^s = \varphi_s^s - \hat{\varepsilon}_s^s \tag{3.17}$$

Where: φ_s^s is the angle between the current and $\underline{a}^s$
$\hat{\varepsilon}_s^s$ is the angle between the $\underline{\hat{\psi}}_s$ and $\underline{a}^s$

Once the angle between the current and stator flux vector is found, it is possible to decompose the current vector in the stator-flux oriented system.

$$i_{s\alpha}^{\psi_s} = \underline{i}_s \cdot \cos\hat{\varphi}_s^s \qquad\qquad i_{s\beta}^{\psi_s} = \underline{i}_s \cdot \sin\hat{\varphi}_s^s \tag{3.18}$$

From Figure 3.5 it can be seen that the angle between stator and rotor flux can be found with help of trigonometry.

$$\hat{\varepsilon}_r^s = \tan^{-1}\left(\frac{x_\sigma \cdot i_{s\beta}^{\psi_s}}{\hat{\psi}_s - x_\sigma \cdot i_{s\alpha}^{\psi_s}}\right) \tag{3.19}$$

The angle between rotor-flux and $\underline{a}^s$-axis and consequently the position of the rotor-flux oriented system can now be calculated. This is done with help of stator-flux angle.

$$\hat{\varepsilon}_r^s = \hat{\varepsilon}_s^s - \hat{\varepsilon}_s^r \tag{3.20}$$

The length of the rotor flux can be found by decomposing the stator flux and current vector into the new rotor-flux oriented system. Rotor flux length is then given by:

$$\psi_R = \psi_{s\alpha}^{\psi_r} - x_\sigma \cdot i_{s\alpha}^{\psi_r} \tag{3.21}$$

3.2.3 Torque equation

The expression for stator flux presented in eq. (3.15) can be decomposed as follows:

$$\psi_{s\alpha}^k = \psi_{R\alpha}^k + x_\sigma \cdot i_{s\alpha}^k \qquad\qquad \psi_{s\beta}^k = \psi_{R\beta}^k + x_\sigma \cdot i_{s\beta}^k \tag{3.22}$$

Electrical torque as a function of stator current and rotor-flux can be found by substituting for stator flux in torque expression shown in eq.(2.37) with help of eq.(3.22).

$$\begin{aligned} \tau_e &= \psi_{s\alpha}^k \cdot i_{s\beta}^k - \psi_{s\beta}^k \cdot i_{s\alpha}^k \\ &= (\psi_{R\alpha}^k + x_\sigma \cdot i_{s\alpha}^k) \cdot i_{s\beta}^k - \left(\psi_{R\beta}^k + x_\sigma \cdot i_{s\beta}^k\right) \cdot i_{s\alpha}^k \\ &= \psi_{R\alpha}^k \cdot i_{s\beta}^k - \psi_{R\beta}^k \cdot i_{s\alpha}^k + x_\sigma\left(i_{s\alpha}^k i_{s\beta}^k - i_{s\beta}^k i_{s\alpha}^k\right) \\ &= \psi_{R\alpha}^k \cdot i_{s\beta}^k - \psi_{R\beta}^k \cdot i_{s\alpha}^k \end{aligned} \tag{3.23}$$

By fixing the system to the rotor-flux and using the relations given in (3.2), electrical torque equation simply becomes:

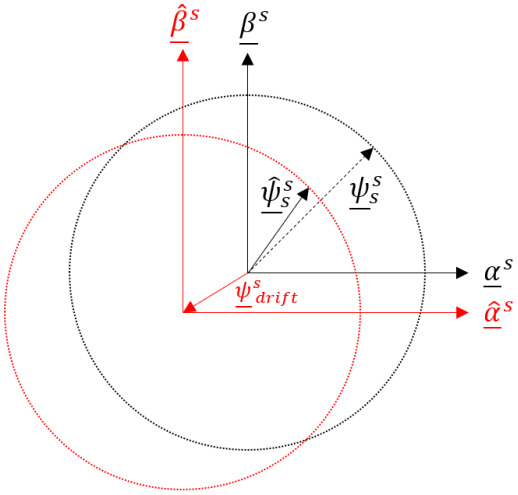
$$\tau_e = \psi_R \cdot i_{s\beta}^{\psi_r} \tag{3.24}$$

As mentioned before, rotor-flux oriented control is chosen in order to get a control strategy similar to the simple DC-machine control. In DC-machine control, field-voltage is used to control the slow magnetization process and armature current is controlled to give a quick torque control. It is seen from eq.(3.11) that change in rotor flux is a slow process due to the large rotor time constant, similar to the change in magnetization. This shows strong resemblance between field voltage control and $i_{s\alpha}^{\psi_r}$ control. From eq.(3.24) it is seen that a change in $i_{s\beta}^{\psi_r}$ will cause an immediate change in torque. From this, it is evident that control of $i_{s\beta}^{\psi_r}$ and armature current can also be treated similarly. This shows that rotor-flux oriented control can be controlled with similar control strategy as the DC-machine.

3.3 Drifting in Voltage Model

Drifting phenomenon is a well know problem related to flux estimations using the voltage model. The reason why drifting only effects the voltage model is because of the open integrator used to calculate stator flux. The problem with an open integrator is that faults in the calculation keeps adding up, introducing a DC-component to the flux estimation. There are several different factors that contributes to drifting. Problems regarding stator voltage estimation due to non-liner voltage losses in the inverter, wrongly estimated stator resistance, offset in current measurement and unsymmetrical physical stator windings may be some of the reasons [17]. Drifting can occur in both dynamic and steady state operation.

The phenomenon itself can be illustrated with help of Figure 3.6. The figure shows two stator-oriented α^s, β^s-planes. The black reference frame illustrates the correct α^s, β^s-plane that is given by the real parameters of the machine and the red reference frame illustrates an estimated $\hat{\alpha}^s, \hat{\beta}^s$-plane, that has been subjected to drifting. If measurements and parameter estimation are done correctly, estimated and real reference frame will coincide, thus making estimated stator flux, $\underline{\hat{\psi}}_s^s$, equal to the correctly calculated stator flux, $\underline{\psi}_s^s$. As the estimation gets subjected to inaccurate measurements or parameter estimates, a DC-component stat to accumulate in the estimated flux, drifting the estimated reference frame away from its correct position.



$$\underline{\hat{\psi}}_s = \underline{\psi}_s^s + \underline{\psi}_{drift}^s$$

Figure 3.6: Drifting of stator flux

It is illustrated in Figure 3.6 that $\underline{\hat{\psi}}_s^s$ follows the circular trajectory of the estimated reference frame. This is because the controller uses coordinates from the estimated reference frame to perform the inverse Park and Clarke transformation when calculating control voltages for the modulator. Even though the trajectory of the vector changes, starting point of the vector, which is given by the physical machine, is assumed to be constant. Analysing the figure, it is observed that as $\underline{\hat{\psi}}_s^s$ rotates anticlockwise, amplitude will oscillate between being smaller and larger than $\underline{\psi}_s^s$. The angle of $\underline{\hat{\psi}}_s^s$ in relation to $\underline{\alpha}^s$-axis will also alternate between leading and lagging $\underline{\psi}_s^s$. Furthermore, an increase in $\underline{\psi}_{drift}^s$ will cause an increase in both oscillation amplitude and angle estimation error. Constant oscillations indicate nether increase or decrees in $\underline{\psi}_{drift}^s$.

To understand how different errors contributes to $\underline{\psi}^s_{drift}$, the expression for $\underline{\hat{\psi}}^s_s$, given by eq.(3.12), can be further developed. The effect of errors can be analysed by dividing estimated and measured parameters into two terms, where one is describing the correct value and one is describing the error value. A similar analysis was presented in the author's project thesis [9]. Consequently, if the measurement or estimate is correct, the error becomes equal to zero. " * " denotes the correct values and " ′ " denotes the error. Expressions for measured voltage, $\underline{u}_s$, measured current, $\underline{i}_s$, and estimated stator resistance, $\hat{r}_s$, are shown below.

$$\underline{u}_s = \underline{u}^*_s + \underline{u}'_s \qquad \underline{i}_s = \underline{i}^*_s + \underline{i}'_s \qquad \hat{r}_s = r^*_s + r'_s \tag{3.25}$$

Using the new notation, an expression describing the correct estimated stator flux can be given as:

$$\underline{u}^*_s = r^*_s \cdot \underline{i}^*_s + \frac{1}{\omega_n} \cdot \frac{d\underline{\psi}^*_s}{dt} \tag{3.26}$$

By introducing the new relations from (3.25) in eq.(3.12), $\underline{\hat{\psi}}^s_s$ can be expressed as:

$$\underline{\hat{\psi}}_s = \omega_n \int_{t_0}^{t} \left(\left(\underline{u}^*_s + \underline{u}'_s\right) - \left(r^*_s + r'_s\right) \cdot \left(\underline{i}^*_s + \underline{i}'_s\right) \right) dt + \underline{\hat{\psi}}_s(t_0) \tag{3.27}$$

Where: $\underline{\hat{\psi}}_s(t_0)$ is the stator flux before magnetization and is equal to zero

The estimated stator flux equation can be further developed by inserting eq.(3.26).

$$\begin{aligned}
\underline{\hat{\psi}}_s &= \omega_n \int_{t_0}^{t} \left(\left(r^*_s \cdot \underline{i}^*_s + \frac{1}{\omega_n} \cdot \frac{d\underline{\psi}^*_s}{dt} + \underline{u}'_s \right) - \left(r^*_s + r'_s\right) \cdot \left(\underline{i}^*_s + \underline{i}'_s\right) \right) dt \\
&= \underline{\hat{\psi}}^*_s + \omega_n \int_{t_0}^{t} \left(\underline{u}'_s + r^*_s \cdot \underline{i}^*_s - r^*_s \cdot \underline{i}^*_s - \left(r^*_s + r'_s\right) \cdot \underline{i}'_s - r'_s \cdot \underline{i}^*_s \right) dt \\
&= \underline{\hat{\psi}}^*_s + \omega_n \int_{t_0}^{t} \left(\underline{u}'_s - \hat{r}_s \cdot \underline{i}'_s - r'_s \cdot \underline{i}^*_s \right) dt \\
&= \underline{\hat{\psi}}^*_s + \underline{\hat{\psi}}'_s
\end{aligned} \tag{3.28}$$

Where: $\underline{\hat{\psi}}^*_s$ is the correctly estimated stator flux
$\underline{\hat{\psi}}'_s$is the error component

Eq.(3.28) shows the three main contributors to the drifting phenomenon. Voltage measurement errors, given by $\underline{u}'_s$, current measurement error, given by $\hat{r}_s \cdot \underline{i}'_s$ and stator resistance estimation error, given by $r'_s \cdot \underline{i}^*_s$. There are several important observations that can be done when analysing the equation. As mentioned earlier, drifting may occur during both steady state and dynamic operation. It can be seen from the equation that all error terms contain a vector. It is the nature of the vector that determines how estimation or measuring faults affect the stator flux.

A common problem when dealing with measurements, is an undesired DC-offset in the measurement. If the flux model is subjected to a DC-offset, it can be seen from eq.(3.28) that the offset vector would cause an incremental increase in $\underline{\hat{\psi}}'_s$ for each sample. The integration of DC-component in $\underline{\hat{\psi}}'_s$ will occur during both steady state and dynamic operation. It is easy to understand that if a DC-offset error is not corrected, stator-flux estimation will over time drift so far away that the control becomes unstable. Since the error leads to a constant increase in drifting it may cause instability during all speed regions of the machine. However, the effect of drifting is most predominant in the low speed area. DC-offsets are usually a result of an unbalanced supply voltage to components in the measuring circuit [18].

Drifting errors caused by stator resistance estimation, would cause drifting during dynamic operation. Stator resistance estimation errors are most problematic during very low speed operation and zero crossings. Electrical conductance of copper is strongly dependent on temperature, thus making stator resistance dependent of operation temperature. To minimise size and cost, induction machines are designed to increase in temperature during nominal load operation. Consequently, stator resistance measured during an identification run with no-load, will not coincide with the actual stator resistance in nominal load operation. It exists methods of on-line adaption of the stator resistance, as presented in [19], but this has not been a part of the scope for this thesis.

Errors in flux estimation due to resistance estimation error is given by the integration of $r_s' \cdot \underline{i}_s^*$. In steady state, $\underline{i}_s^*$ is purely sinusoidal and no DC-offset is integrated. However, during magnetisation for example, $\underline{i}_s^*$ becomes a DC-vector, causing drifting similarly to DC-offsets in measurements. It is worth mentioning that even though resistance estimation errors do not contribute to drifting during steady state operation, the error will still lead to a wrongly estimated stator-flux amplitude. A thorough analysis of rotor resistance estimation error's effect on the voltage model was presented in the master thesis [7].

It exists many different correction methods that tackles the problems regarding drifting. The correction methods use a so-called observer in order to compensate for the different errors. Drift compensation is added before the integration of stator flux. Even though observers are able to compensate for drifting, it is still important to put effort into getting correct measurements and parameter estimations.

Correction observers has been a topic in previous doctor, master and project thesis at NTNU. A comparison between a Closed Loop observer with a P-regulator in the feedback loop [3] and the Niemelä observer [5] was done in [7]. The Closed Loop observer was further studied in the author's project thesis [9] where difference between Closed Loop observers with P-regulator and PI-regulator [4] were analysed. The Closed Loop observer that was analysed in the author's project thesis used the voltage model to estimate a stator flux amplitude and angle. Estimated stator flux frequency and current measurement were fed into the current model and an alternative stator flux estimation was calculated. Current model did not suffer from drifting and was therefore used in the lower speed areas. As the speed increased parameter estimation had a lower impact on the voltage model, thus making it the most desirable estimation method. The regulator in the feedback loop forced the voltage model to follow estimation from the current model at very low speeds while gradually removing its influence as the machine accelerated.

4 Voltage and Current Measurement

4.1 Non-ideal Inverter Model

The two-level three-phase inverter is a common topology when controlling an induction machine. The inverter consists of three bridge legs, each with two IGBTs and two power diodes connected in anti-parallel. Inverter topology is presented in Figure 4.1. When the inverter is controlled by a *pulse-width-modulator* (PWM) output phase voltage is controlled by comparing a sinusoidal control voltage to a constant triangular wave. In an ideal converter, ratio between control voltage and triangular wave matches the ratio between output phase voltage and DC-link voltage. However, non-ideal properties of the semi-conducting components introduce a non-linear voltage drop on the inverters output terminals.

In this subchapter it will be shown how the non-ideal inverter can be modelled. The model presented is the well-known inverter model first deducted in [20].The goal of the model is to find a voltage expression that contain the actual output voltage, the non-linear voltage drop and the resistive voltage drop of the inverter.

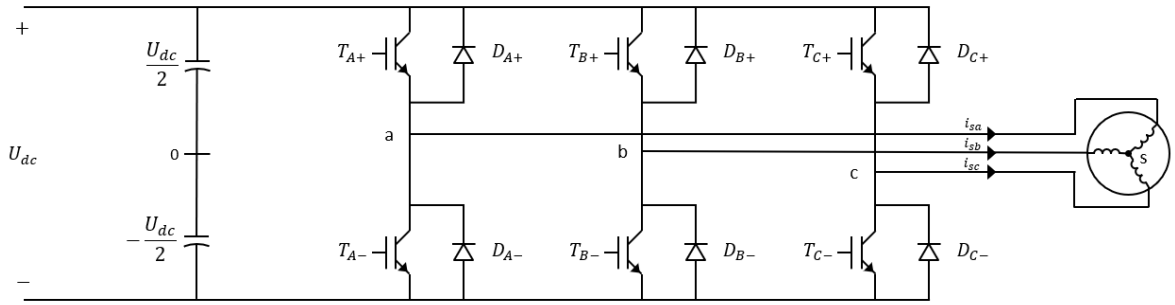


Figure 4.1: Inverter circuit

4.1.1 Switching characteristics

Switching characteristics of the IGBT itself is one of the contributing factors to non-linear voltage drop in the inverter. The IGBT has a time delay from it receives a turn-on signal until the bridge leg midpoint has approximately the same voltage potential as the DC-terminal to which it is connected, and vice versa. Turn-on waveform is presented in Figure 4.2. The first time-delay, $t_{d(on)}$, is the time needed for gate-source voltage, U_{GS}, to rise from zero to threshold voltage, $U_{GS(th)}$. Experimental tests conducted in [21] shows that $t_{d(on)}$ decreases with increasing temperatures. Once U_{GS} reaches $U_{GS(th)}$, drain current starts to increase according to the voltage-current transfer curve of the IGBT. Current through the IGBT rises alongside with U_{GS} until it carries the entire load current. This rise time is noted t_{ri} in the figure. Once the IGBT carries full load current, U_{GS} is clamped to its respective value, known in the literature as the Miller plateau, and the voltage over the component starts to drop. It is seen that the voltage drops with two different gradients. t_{fv1} corresponds to the active region and t_{fv2} corresponds to the ohmic region in the voltage-current output characteristics [22]. From Figure 4.2 it is seen that time intervals $t_{d(on)}$, t_{ri} and t_{fv1} are the one of interest when analysing the time delay from turn-on signal to the midpoint has the correct voltage potential. Furthermore, this time interval decreases with rising temperatures and increase with increasing load currents.

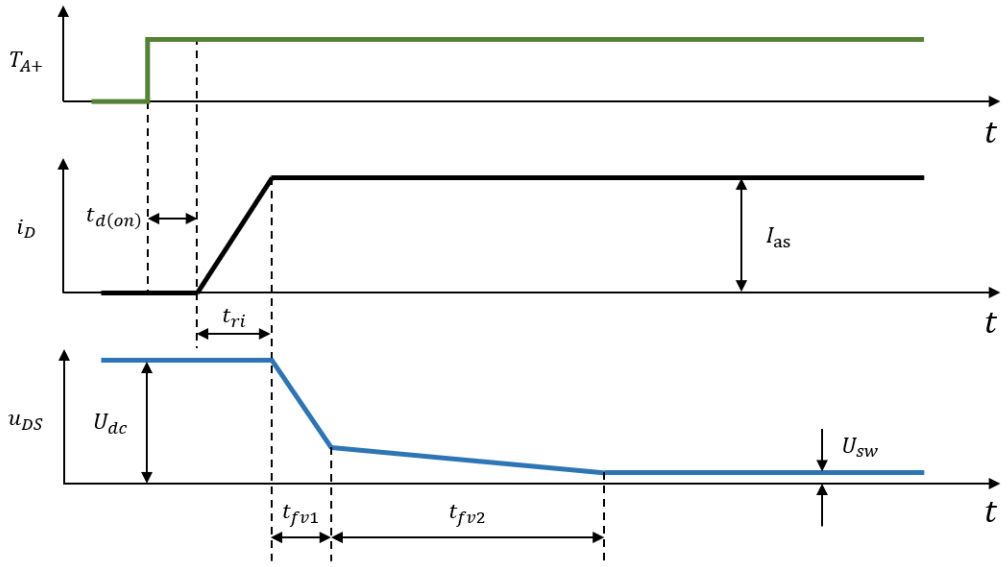


Figure 4.2: Turn-on voltage and current waveform

Turn-off waveform of the IGBT is presented in Figure 4.3 and is the inverse sequence of the turn-on waveform. Initially in the closed IGBT, U_{GS} is equal to the gate voltage U_{GG}. When the IGBT is turned off, U_{GS} has to be reduced to the Miller plateau before the voltage drop over the component start to increase. This time-delay is called $t_{d(off)}$. The Miller plateau is proportional with load current. Consequently, small load currents result in longer $t_{d(off)}$ delay and vice versa. Research presented in [21] shows that $t_{d(off)}$ increases with increased temperatures. Once the gate-source voltage reaches the Miller plateau, voltage drop over the IGBT will increase until it is equal to the DC-link voltage. The voltage rise time is called t_{rv}. From a voltage potential perspective, it is seen that the time from a turn-off signal is given until the IGBT is blocking the voltage is given by $t_{d(off)}$ and t_{rv}.

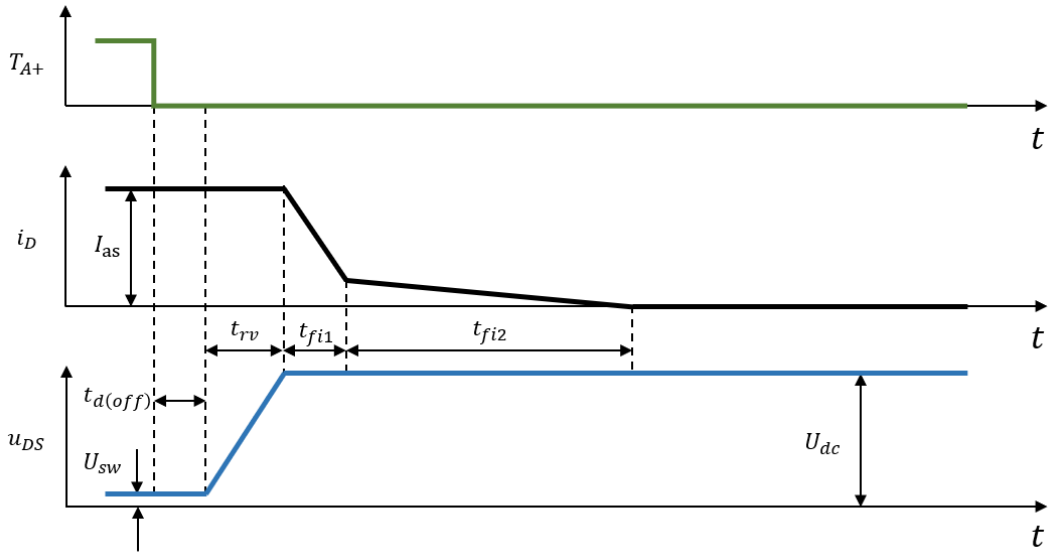


Figure 4.3: Turn-off voltage and current waveform

4.1.2 Blanking time and on/off time

The switching patterns for inverter leg A is shown in Figure 4.4. It is assumed that the current is flowing to the load. Figure 4.4-(a) shows the switching pattern sent from the modulator, i.e. without any time delays. The upper IGBT turns on at the same time instance as the lower IGBT turns off.

However, in real life inverters, it is necessary to introduce a time delay between the turn-off and turn-on of the IGBTs in order to prevent a short circuit of the bridge leg. This time delay is called *blanking time*, Δt, and is shown in Figure 4.4-(b).

Furthermore, as shown in previous section, the IGBT itself introduce a delay between the initial turn-on and-off signal and changed in midpoint potential. Figure 4.4-(c) shows the phase-to-0 voltage, U_{a0}, of the inverter. From previous section it is seen that rising time is given by $t_{d(on)} + t_{ri} + t_{fv1}$ and falling time is given by $t_{d(off)} + t_{rv}$. To simplify the expression of U_{a0}, the voltage can be expressed as in Figure 4.4-(d).

From Figure 4.4-(a), on-time from the modulator, T_a is $T_2 - T_1$. However, looking at the equivalent U_{a0} it is evident that this is not true. It can be seen from the figure that on-time describing the midpoints voltage potential is given by $T_2 - T_1 - \Delta t - t_{on} + t_{off}$. By comparing the two on-times, the error between them can be expressed as:

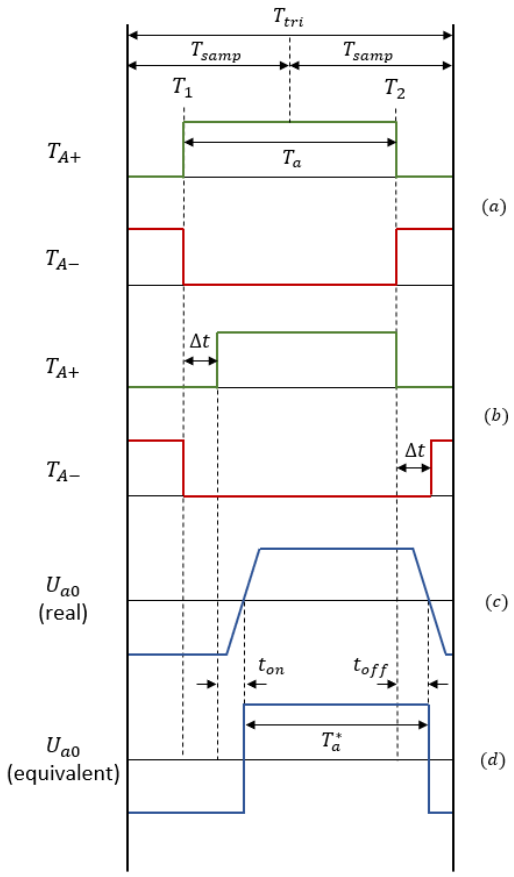


Figure 4.4: Switching patterns for a two level three-phase inverter with current flowing to the load. (a) Ideal switching. (b) Switching this blanking time. (c) Actual phase to 0 voltage. (d) equivalent phase to 0 voltage

$$T_{err} = T_a - T_a^* = (T_2 - T_1) - (T_2 - T_1 - \Delta t - t_{on} + t_{off}) = t_{off} - t_{on} - \Delta t \quad (4.1)$$

Where $t_{on} = t_{d(on)} + t_{ri} + t_{fv1}/2$

$t_{off} = t_{d(off)} + t_{rv}/2$

When current in the inverter switches direction, it can be shown [20] that T_{err} can be expressed as:

$$T_{err} = -(t_{off} - t_{on} - \Delta t) \quad (4.2)$$

The two time-error equations can be generalized by introducing a current direction dependency

$$t' = \text{sing}(i_{sa})(t_{off} - t_{on} - \Delta t) \quad (4.3)$$

Where $\text{sing}(i_{an}) = \begin{cases} 1: & i_{sa} > 0 \\ -1: & i_{sa} < 0 \end{cases}$

The on-time of T_{A+} given by the modulator and shown in Figure 4.4-(a), can be expressed by midpoint voltage on-time and the time-error.

$$T_a = T_a^* + \text{sign}(i_{sa}) \cdot t' \tag{4.4}$$

Where: T_a^* is the equivalent on-time shown in Figure 4.4 (d)
$t' = t_{off} - t_{on} - \Delta t$

Doing the same analysis for the two other phases, gives the same result. On-times can be described by the following equations.

$$T_b = T_b^* + \text{sign}(i_{sb}) \cdot t' \tag{4.5}$$

$$T_c = T_c^* + \text{sign}(i_{sc}) \cdot t' \tag{4.6}$$

It is important to note that the duration of turn-on and -off time delays are not constant but dependent on load current. Furthermore, turn-off time is usually much longer than turn-on time. The difference can be in the hundreds of nanoseconds range [23].

4.1.3 Voltage drop over the IGBT and power diode

Blanking time, on-time and off-time are not the only contributors to reduction in phase voltage in the inverter. Both power diodes and IGBTs will introduce a voltage drop due to its non-ideal properties. This voltage drop is called f*orward voltage*. Forward voltage can be divided into a constant voltage drop and a current dependent voltage drop for both devices. The constant voltage drop is a result of the properties of a semiconductor and the current dependent voltage drop is given by the equivalent on-state resistance.

The forward voltage drop of the IGBT and power diode are given by eq.(4.7) and (4.8), respectively.

$$U_{sw} = U_{sw0} + r_{sw} \cdot |i_{sa}| \tag{4.7}$$

$$U_{fd} = U_{fd0} + r_{fd} \cdot |i_{sa}| \tag{4.8}$$

Where: U_{sw0} is the constant voltage drop over the IGBT switch
U_{fd0} is the constant voltage drop over the power diode
r_{sw} is the equivalent on-state resistance of the IGBT switch
r_{fd} is the equivalent on-state resistance of the power diode

Analysing Figure 4.1 it can be found that the instantaneous phase-to-zero voltage, u_{a0}, is affected by both current direction and on/off status of the IGBTs within the bridge leg. Given the two possible current directions and the complementary switching of the IGBTs, it is evident that the voltage-drop in u_{a0} can be affected by the forward voltage in four different ways.

When the current is flowing in positive direction, i.e. from the DC-link to the load, it can be seen that the current flows through the upper IGBT when T_{A+} is on and through the lower freewheeling power diode when T_{A-} is on. u_{a0} given the two switching states and positive current direction are presented in eq.(4.9) and (4.10).

$$u_{a0} = \frac{U_{dc}}{2} - U_{sw} \qquad (T_{A+} \text{ is on and } T_{A-} \text{ is off}) \tag{4.9}$$

$$u_{a0} = -\frac{U_{dc}}{2} - U_{fd} \qquad (T_{A-} \text{ is on and } T_{A+} \text{ is off}) \tag{4.10}$$

By introducing the variable S_A that defines which of the two IGBTs that are conducting, eq.(4.9) and (4.10) can be combined into the following equation.

$$u_{a0} = \left(U_d - U_{sw} + U_{fd}\right) \cdot \left(S_A - \frac{1}{2}\right) - \frac{1}{2} \cdot \left(U_{sw} + U_{fd}\right) \tag{4.11}$$

Where: $S_A = 1$: (T_{A+} is on and T_{A-} is off)
$S_A = 0$: (T_{A-} is on and T_{A+} is off)

When the current direction is reversed, current will flow through the upper diode when T_{A+} and through the lower IGBT when T_{A-} is on. u_{a0} is then given by eq.(4.12) and (4.13).

$$u_{a0} = \frac{U_{dc}}{2} + U_{fd} \qquad (T_{A+} \text{ is on and } T_{A-} \text{ is off}) \tag{4.12}$$

$$u_{a0} = -\frac{U_{dc}}{2} + U_{sw} \qquad (T_{A-} \text{ is on and } T_{A+} \text{ is off}) \tag{4.13}$$

Similarly as with positive current direction, eq.(4.12) and (4.13) can be expressed as a function of the on/off state of the switches in the bridge leg.

$$u_{a0} = \left(U_{dc} - U_{sw} + U_{fd}\right) \cdot \left(S_A - \frac{1}{2}\right) + \frac{1}{2}\left(U_{sw} + U_{fd}\right) \tag{4.14}$$

Since the switching in the inverter has a much higher frequency than the fundamental frequency, it can be assumed that the current diction does not change during one switching period. Using this assumption and by introducing the sign of the respective bridge leg current, it is possible to express eq.(4.11) and (4.14) by the following equation.

$$u_{a0} = \left(U_{dc} - U_{sw} + U_{fd}\right) \cdot \left(S_A - \frac{1}{2}\right) - \frac{1}{2}\text{sign}(i_{sa}) \cdot \left(U_{sw} + U_{fd}\right) \tag{4.15}$$

By inserting eq.(4.7) and (4.8) in eq.(4.15), u_{a0} can be expressed as a function of the constant and current dependent voltage drop of the semiconducting components in the circuit.

$$\begin{aligned} u_{a0} &= \left(U_{dc} - U_{sw} + U_{fd}\right) \cdot \left(S_A - \frac{1}{2}\right) - \frac{1}{2}\text{sign}(i_{sa}) \cdot \left(U_{sw0} + r_{sw} \cdot |i_{sa}| + U_{fd0} + r_{fd} \cdot |i_{sa}|\right) \\ &= \left(U_{dc} - U_{sw} + U_{fd}\right) \cdot \left(S_A - \frac{1}{2}\right) - \frac{1}{2}\text{sign}(i_{sa}) \cdot \left(U_{sw0} + U_{th,fd0}\right) - \frac{1}{2}\left(r_{sw} + r_{fd}\right) \cdot i_{sa} \end{aligned} \tag{4.16}$$

The mean phase-to-zero voltage, U_{a0}, can be found by substituting S_A with bridge leg on-time and dividing it by the switching period, or two times the sampling time, when using synchronised PWM.

$$U_{a0} = \left(U_{dc} - U_{sw} + U_{fd}\right)\left(\frac{T_a}{2T_{samp}} - \frac{1}{2}\right) - \frac{1}{2}\text{sign}(i_{sa})\left(U_{sw0} + U_{fd0}\right) - \frac{1}{2}\left(r_{sw} + r_{fd}\right) \cdot i_{sa} \tag{4.17}$$

When conducting the same analysis on the other phases, similar results are found.

$$U_{a0} = \left(U_{dc} - U_{sw} + U_{fd}\right)\left(\frac{T_b}{2T_{samp}} - \frac{1}{2}\right) - \frac{1}{2}\text{sign}(i_{sb})\left(U_{sw0} + U_{fd0}\right) - \frac{1}{2}\left(r_{sw} + r_{fd}\right) \cdot i_{sb} \tag{4.18}$$

$$U_{c0} = \left(U_{dc} - U_{sw} + U_{fd}\right)\left(\frac{T_c}{2T_{samp}} - \frac{1}{2}\right) - \frac{1}{2}\text{sign}(i_{sc})\left(U_{sw0} + U_{fd0}\right) - \frac{1}{2}\left(r_{sw} + r_{fd}\right) \cdot i_{sc} \tag{4.19}$$

4.1.4 Phase voltage of the machine

The voltage between bridge leg midpoint and neutral point of the machine for the three different phases can be expressed by eq.(4.20)-(4.22). The equations show that the phase-to-neutral voltage is given by its respective phase-to-zero voltage and a neutral-to-zero voltage, U_{s0}. Note that U_{s0} is equal in all the phase voltages.

$$U_{sa} = U_{a0} - U_{s0} \tag{4.20}$$

$$U_{sb} = U_{b0} - U_{s0} \tag{4.21}$$

$$U_{sc} = U_{c0} - U_{s0} \tag{4.22}$$

According to Kirchhoff's first law, the total sum of currents entering a junction is equal to the total sum leaving the junction. Since the neutral point of the machine is isolated from ground, the sum of currents entering the neutral has to be zero.

$$i_{sa} + i_{sb} + i_{sc} = 0 \tag{4.23}$$

The three-phased motor connected to the inverter is assumed to be a balanced load. Thus, the sum phase-voltages are also zero.

$$U_{sa} + U_{sb} + U_{sc} = 0 \tag{4.24}$$

An expression of U_{s0} can be found by inserting eq.(4.20)-(4.22) in (4.24).

$$\begin{aligned} &U_{a0} + U_{b0} + U_{c0} - 3U_{s0} = 0 \\ &U_{s0} = \frac{1}{3}(U_{a0} + U_{b0} + U_{c0}) \end{aligned} \tag{4.25}$$

The expression can be further developed by inserting eq.(4.17)-(4.19).

$$\begin{aligned} U_{s0} = \frac{1}{3}\Bigg(&\big(U_{dc} - U_{sw} + U_{fd}\big)\left(\left(\frac{T_a}{2T_{samp}} - \frac{1}{2}\right) + \left(\frac{T_b}{2T_{samp}} - \frac{1}{2}\right) + \left(\frac{T_c}{2T_{samp}} - \frac{1}{2}\right)\right)\Bigg) \\ -\frac{1}{3}\Bigg(&\frac{1}{2}\big(U_{sw0} + U_{fd0}\big)\big(\mathrm{sign}(i_{sa}) + \mathrm{sign}(i_{sb}) + \mathrm{sign}(i_{sc})\big) - \frac{1}{2}\big(r_{sw} + r_{fd}\big)\cdot(i_{sa} + i_{sb} + i_{sc})\Bigg) \end{aligned} \tag{4.26}$$

By realizing that the sum of currents is equal to zero eq.(4.26) simplifies to:

$$\begin{aligned} U_{n0} = &\frac{1}{3}\big(U_{dc} - U_{sw} + U_{fd}\big)\left(\frac{T_a + T_b + T_c}{2T_{samp}} - \frac{3}{2}\right) \\ &-\frac{1}{6}\big(U_{sw0} + U_{fd0}\big)\big(\mathrm{sign}(i_{sa}) + \mathrm{sign}(i_{sa}) + \mathrm{sign}(i_{sa})\big) \end{aligned} \tag{4.27}$$

Going back to eq.(4.20) and inserting eq.(4.17) and (4.27), the expression for U_{sa} becomes:

$$\begin{aligned} U_{sa} &= U_{a0} - U_{s0} \\ &= \big(U_{dc} - U_{sw} + U_{fd}\big)\left(\left(\frac{T_a}{2T_{samp}} - \frac{1}{2}\right) - \frac{1}{3}\left(\frac{T_a + T_b + T_c}{2T_{samp}} - \frac{3}{2}\right)\right) \\ &-\big(U_{sw0} + U_{fd0}\big)\left(\frac{1}{2}\mathrm{sign}(i_{sa}) - \frac{1}{6}\big(\mathrm{sign}(i_{sa}) + \mathrm{sign}(i_{sb}) + \mathrm{sign}(i_{sc})\big)\right) \\ &-\frac{1}{2}\big(r_{sw} + r_{fd}\big)\cdot i_{sa} \end{aligned} \tag{4.28}$$

With help of some simple algebraic manipulation, eq.(2.28) can be expressed as follows:

$$\begin{aligned} U_{sa} = &\frac{1}{3}\left(U_{dc} - U_{sw} + U_{fd}\right)\left(\frac{2T_a - T_b - T_c}{2T_{samp}}\right) \\ &-\frac{1}{6}\left(U_{sw0} + U_{fd0}\right)\left(2\text{sign}(i_{sa}) - \text{sign}(i_{sb}) - \text{sign}(i_{sc})\right) \\ &-\frac{1}{2}\left(r_{sw} + r_{fd}\right)\cdot i_{sa} \end{aligned} \tag{4.29}$$

Similar analysis can be done for the other two phases.

$$\begin{aligned} U_{sb} = &\frac{1}{3}\left(U_{dc} - U_{sw} + U_{fd}\right)\left(\frac{2T_b - T_a - T_c}{2T_{samp}}\right) \\ &-\frac{1}{6}\left(U_{sw0} + U_{fd0}\right)\left(2\text{sign}(i_{sb}) - \text{sign}(i_{sa}) - \text{sign}(i_{sc})\right) \\ &-\frac{1}{2}\left(r_{sw} + r_{fd}\right)\cdot i_{sb} \end{aligned} \tag{4.30}$$

$$\begin{aligned} U_{sc} = &\frac{1}{3}\left(U_{dc} - U_{sw} + U_{fd}\right)\left(\frac{2T_c - T_a - T_b}{2T_{samp}}\right) \\ &-\frac{1}{6}\left(U_{sw0} + U_{fd0}\right)\left(2\text{sign}(i_{sc}) - \text{sign}(i_{sa}) - \text{sign}(i_{sb})\right) \\ &-\frac{1}{2}\left(r_{sw} + r_{fd}\right)\cdot i_{sc} \end{aligned} \tag{4.31}$$

Eq.(4.29)-(4.31) express the different voltage components in the phase-voltages. However, the equations do not yet differentiate the output voltage of the inverter from and the non-linear voltage drop. Going back to section 4.1.2, it can be seen that the on-time used in the phase-voltage equation, represents on-times from the modulator. By inserting eq.(4.4)-(4.6), the equation can be further extended.

$$\begin{aligned} U_{sa} = &\frac{1}{3}\left(U_{dc} - U_{sw} + U_{fd}\right)\left(\frac{2T_a^* - T_b^* - T_c^*}{2T_{samp}} - \frac{2t'\cdot\text{sign}(i_{sa}) - t'\cdot\text{sign}(i_{sb}) - t'\cdot\text{sign}(i_{sc})}{2T_{samp}}\right) \\ &-\frac{1}{6}\left(U_{sw0} + U_{fd0}\right)\left(2\text{sign}(i_{sa}) - \text{sign}(i_{sb}) - \text{sign}(i_{sc})\right) - \frac{1}{2}\left(r_{sw} + r_{fd}\right)\cdot i_{sa} \end{aligned} \tag{4.32}$$

The description of U_{sa} given by eq. (4.32) includes actual on-time, time error and voltage drop caused by the semiconducting components, both linear and non-linear. Using the different elements of the equation, it is possible to express the phase-voltage, U_{sa}, in terms of output voltage, distorted voltage and resistive voltage drop.

$$U_{sa} = U_{sa}^* + U_{sa}' - \frac{1}{2}\left(r_{th,sw} + r_{th,fd}\right)\cdot i_{sa} \tag{4.33}$$

The output voltage is represented as U_{sa}^* and is given by the DC-link voltage, total voltage drops of the IGBT and power diode and the on-time describing electrical potential at the bridge leg. The voltage drops of the semiconducting components are much smaller than the DC-link voltage and can therefore be neglected.

$$\begin{aligned} U_{sa}^* &= \frac{1}{3}\left(U_{dc} - U_{sw} + U_{fd}\right)\left(\frac{2T_a^* - T_b^* - T_c^*}{2T_{samp}}\right) \\ &\approx \frac{1}{3}\cdot U_{dc}\cdot\left(\frac{2T_a^* - T_b^* - T_c^*}{2T_{samp}}\right) \end{aligned} \tag{4.34}$$

The distorted voltage is represented as U'_{sa} and is governed by DC-link voltage, voltage drops caused by the IGBT and power diode, on-time error, and the direction of currents. The approximation regarding DC-link voltage done in eq.(4.34), is also true for U'_{sa}.

$$\begin{aligned} U'_{sa} &= \frac{1}{3}\left(U_{dc} - U_{sw} + U_{fd}\right)\left(\frac{2t' \cdot sign(i_{sa}) - t' \cdot sign(i_{sb}) - t' \cdot sign(i_{sc})}{2T_{samp}}\right) \\ &\quad -\frac{1}{6}\left(U_{sw0} + U_{fd0}\right)\left(2\text{sign}(i_{sa}) - \text{sign}(i_{sb}) - \text{sign}(i_{sc})\right) \\ &= \frac{1}{6}\left(\left(U_{dc} - U_{sw} + U_{fd}\right)\left(\frac{t'}{T_{samp}}\right) - U_{sw0} - U_{fd0}\right)\left(2\text{sign}(i_{sa}) - \text{sign}(i_{sb}) - \text{sign}(i_{sc})\right) \\ &\approx \frac{1}{6}\left(\frac{U_{dc} \cdot t'}{T_{samp}} - U_{sw0} + U_{fd0}\right)\left(2\text{sign}(i_{sa}) - \text{sign}(i_{sb}) - \text{sign}(i_{sc})\right) \end{aligned} \tag{4.35}$$

Analysing the other two phase-to-neutral voltages gives a similar result.

$$U_{sb} = U^*_{sb} + U'_{sb} - \frac{1}{2}\left(r_{sw} + r_{fd}\right) \cdot i_{sb} \tag{4.36}$$

$$U^*_{sb} \approx \frac{1}{3} \cdot U_{dc} \cdot \left(\frac{2T^*_b - T^*_a - T^*_c}{2T_{samp}}\right) \tag{4.37}$$

$$U'_{sb} \approx \frac{1}{6}\left(\frac{U_{dc} \cdot t'}{T_{samp}} - U_{sw0} + U_{fd0}\right)\left(2\text{sign}(i_{sb}) - \text{sign}(i_{sa}) - \text{sign}(i_{sc})\right) \tag{4.38}$$

$$U_{sc} = U^*_{Cn} + U'_{Cn} - \frac{1}{2}\left(r_{sw} + r_{fd}\right) \cdot i_{sc} \tag{4.39}$$

$$U^*_{sc} \approx \frac{1}{3} \cdot U_{dc} \cdot \left(\frac{2T^*_b - T^*_a - T^*_c}{2T_{samp}}\right) \tag{4.40}$$

$$U'_{sc} \approx \frac{1}{6}\left(\frac{U_{dc} \cdot t'}{T_{samp}} - U_{sw0} + U_{fd0}\right)\left(2\text{sign}(i_{sc}) - \text{sign}(i_{sa}) - \text{sign}(i_{sb})\right) \tag{4.41}$$

4.2 Consequences of the Non-linear voltage drop

Previous subchapter showed an expression for the output voltage of the inverter together with its resistive and non-linear voltage drop. These expressions are important in order to understand how the nonlinearity affects the drive, especially when the speed of the machine is low. The distortion caused by the inverter affect open loop V/f-control and closed loop FOC somewhat different. The sinusoidal control voltage used in V/f-control leads to a distorted output voltage. The current controller used in FOC forces the output voltage of the inverter to have a sinusoidal waveform by introducing the same distortion seen in V/f-control in the control voltage, only with opposite sign.

The waveforms and amplitudes of control voltage, distorted voltage and output voltage in a V/f-controlled drive is shown in Figure 4.5. The figure shows the equations deducted in previous subchapter and illustrates voltages in a machine operating at 5% of nominal speed. Figure 4.5-(a) shows the control voltage calculated in the controller. This is the desired output voltage for that given frequency. The boost voltage that was discussed in chapter 3.1 are used to compensate for the resistive voltage drops in the inverter. The control voltage can be described by following equation:

$$U_{control} = \frac{1}{3} \cdot U_{dc} \cdot \left(\frac{2T_a - T_b - T_c}{2T_{samp}} \right)$$

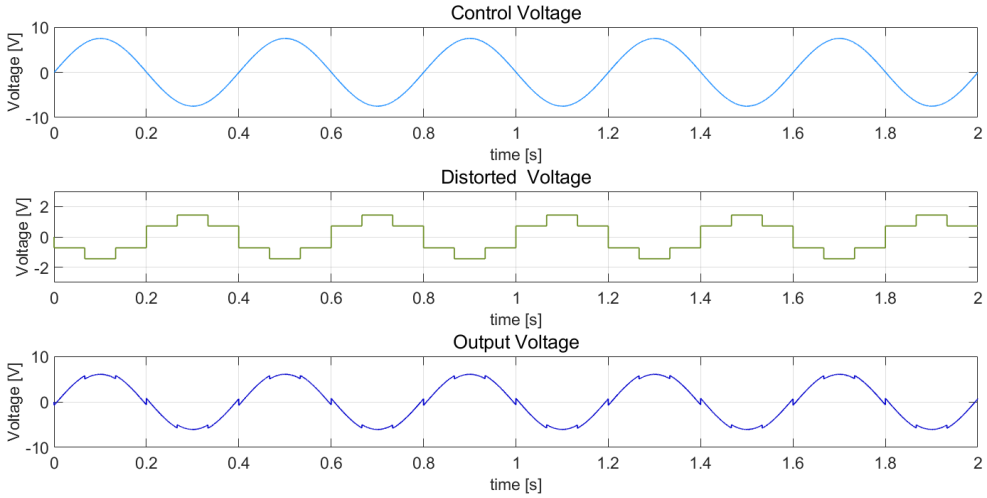


Figure 4.5: V/f-controlled inverter voltages

Figure 4.5-(b) show the distorted voltage U'_{sa}. U'_{sa} can be described by eq.(4.35) and the equation is repeated below. The distorted voltage has a trapezoidal waveform that changes in amplitude as a function of the direction of the phase currents. During the complete time period of one phase current, all three phase currents changes polarity once, making a total of six current directions. Thus, distorted voltage will also change its value six times within the time period. The six-step change in value can also be seen in the figure. Furthermore, eq.(4.35) shows that the amplitude of the distorted voltage is given by DC-link voltage, non-linear component of the forward voltage, sampling time and the time error, t'. Since these values are more or less constant, the distorted voltage can be considered as independent of output current and voltage.

$$U'_{sa} \approx \frac{1}{6} \left(\frac{U_{dc} \cdot t'}{T_{samp}} - U_{th,sw0} + U_{th,fd0} \right) \left(2\text{sign}(i_{sa}) - \text{sign}(i_{sb}) - \text{sign}(i_{sc}) \right)$$

The peak amplitude of U'_{sa} occurs when the current direction of i_{sa} is positive and when i_{sb}, i_{sc} is negative or vice versa. Using eq. (4.35) and inserting the different current directions shows a peak value of $2/3\left(U_{dc} \cdot t'/T_{samp} - U_{th,sw0} + U_{th,fd0}\right)$. Moreover, distorted voltage is always in counter-phase with the respective phase-current.

Figure 4.5-(c) shows output phase voltage, U_{sa}^*. The trapezoidal distortion voltage from Figure 4.5-(b) acts as a non-linear voltage drop on the output voltage. The figure shows that output voltage, when using a V/f-controller, results in a discontinuous voltage with a lower amplitude compared to the control voltage. The equation describing output voltage is presented in eq.(4.34) is repeated below.

$$U_{sa}^* \approx \frac{1}{3} \cdot U_{dc} \cdot \left(\frac{2T_a^* - T_b^* - T_c^*}{2T_{samp}} \right)$$

$$U_{sa}^* = U_{control} + U_{sa}' \tag{4.42}$$

The output voltage can also be expressed by the control voltage and the distortion voltage as shown in eq.(4.42). Looking at the equation, it is easy to understand why the distorted voltage has the largest impact during low speed operation. Control voltage amplitude is proportional with stator frequency and will therefore have a small amplitude at low speeds. Distortion voltage amplitude however, can be assumed to be independent of frequency, thus causing a significant distortion at lower speeds. This effect is illustrated in the figures below. Figure 4.6, Figure 4.7 and Figure 4.8 show the rotating voltage vector in the machine, $\underline{u}_s^s$ with 20%, 5% and 2.5% of nominal speed, respectively. Influence of U_{sa}' is clearly increasing as the machine slows down. The discontinues trajectory of $\underline{u}_s^s$ creates a six-harmonic component that causes toque ripples in the machine if not compensated for.

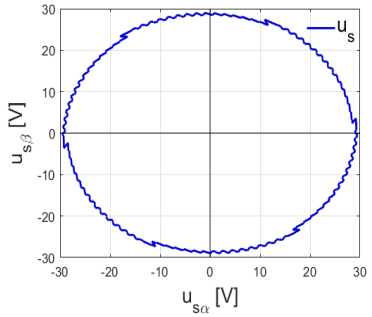


Figure 4.6: 20% speed

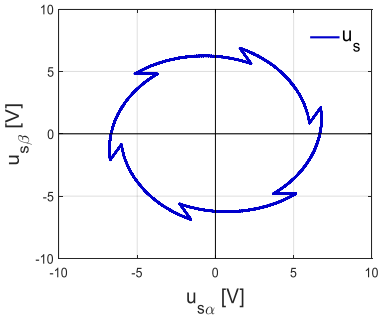


Figure 4.7: 5% speed

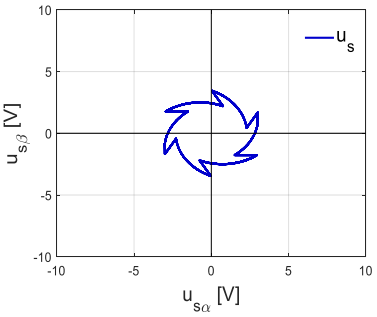


Figure 4.8: 2.5% speed

As mention earlier, non-ideal properties of the inverter can also influence closed loop FOC. The current controller used in FOC regulates the phase current considerably faster than its reference value. The controller is able to keep a sinusoidal output voltage and current by introducing the same distortions shown in V/f-control (Figure 4.5-(b)) in the control voltage. Consequently, the distorted control voltage will contain the same six-harmonic component as the output voltage in open loop control [24].

As shown in section 3.2.2, stator voltage is an input variable when operating in sensorless control. It is possible to acquire knowledge of stator voltage either by direct measurements, indirect measurements, or through estimation. The benefits of estimating the phase voltage is the reduction in required measuring hardware. This reduces both costs and sources of errors. A common method of estimating the stator voltage is to use the control voltage given by the current controller. This simple solution gives an accurate estimate of the actual voltage in medium and high-speed operation. However, as the speed decreases to the lower speed regions and the sixth-harmonic component generated by the current controller becomes more dominant in the control voltage, the accuracy of the estimation also decrees [19]. Moreover, if the control voltage is distorted with a sixth-harmonic component, the calculated fluxes will also be affected by the same harmonic component [24]. The harmonic component in the flux will generate ripples in the electrical torque, just like in open loop control. If the operating speed reduced even further, distorted voltage may exceed the required control voltage in amplitude. This will make it impossible to achieve a stable operation [19]. A common solution to this problem is to use the description of distorted voltage as a feedforward term after the current controller. Consequently, both control voltage and output voltage become sinusoidal. An example of this is shown in [10].

Another method is to estimate output voltage using eq.(4.34). This method requires knowledge of the time interval where the midpoint voltage is equal to U_{dc}. One of the benefits with this method is that it is able to estimate phase voltage of both V/f-controlled and FOC drives during all speed ranges. It is this estimation method that will be implemented into the control design in this thesis.

4.3 Current measurement

In control strategies like Field Oriented Control, correct current measurements are essential in order to ensure stable control. As discussed in chapter 3.3, current measurements errors, together with other estimation and measurement errors will cause a phenomenon called drifting. Drifting gives the estimated flux an oscillatory behaviour and causes oscillations in speed, thus vibrations. It can also make the control itself unstable. Offset errors and scaling errors are the most common problems regarding current measurement. Both errors may contribute to drifting. Furthermore, the errors themselves will also cause vibrations in the machine regardless of drifting. An analysis of current measurement error's effect on FOC AC-machines are presented in [18] and mathematical derivations presented in this subchapter is based in this work. The paper also proposes a correction method for these errors. However, the method relies on having precise speed measurement and the estimated speed obtained in sensorless control does not give high enough accuracy [10].

The consequences of measuring errors are illustrated in Figure 4.9. The figure shows the mechanical speed of a machine controlled by a two-level three-phase inverter without any compensation or correction for errors. It also shows an FFT analysis of the speed. It is evident that the presence of current measurement errors causes oscillation in the speed. Speed oscillations will increase the vibrations in the machine, thus reducing lifetime on mechanical components like bearings, mechanical seals, shaft couplings or gear transition. The FFT analysis shows that the oscillations are mainly governed by three harmonic components equal to (f_e), two times greater ($2f_e$) and six times greater ($6f_e$) than the electrical frequency. The harmonic component with frequency f_e is caused by offset errors and the component with frequency $2f_e$ is caused by scaling errors. The harmonic component of $6f_e$ is a consequence of the nonlinear voltage drop in the inverter, discussed in previous subchapter.

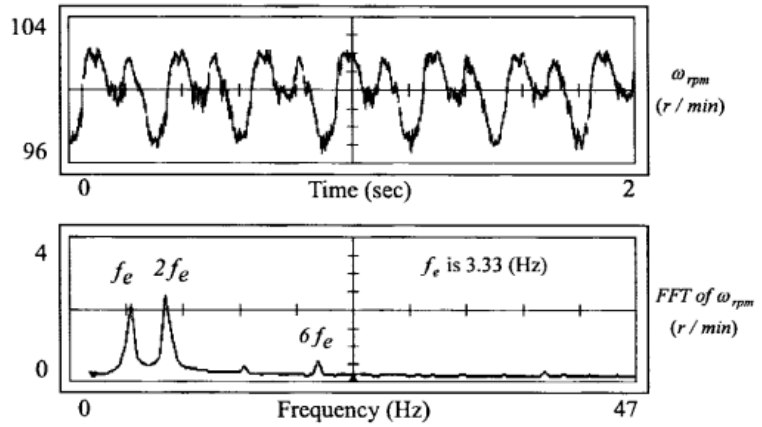


Figure 4.9: Mechanical speed and FFT analysis of mechanical speed with measurement errors [18].

These torque ripples act as a disturbance load on the mechanical system and decrease the performance of the speed controller. It is important to further analyse the reasons for measurement errors, how they affect the measuring signals and how wrongly measured signals affect the motor equations in order to understand and reduce these problems.

4.3.1 Current measurement circuit

An illustration of the measuring circuit is presented in Figure 4.10. The measuring circuit typically consists of a closed loop Hall effect current sensor, a buffer/scaling circuit, an anti-aliasing filter, an AD-converter, a FPGA and a processor.

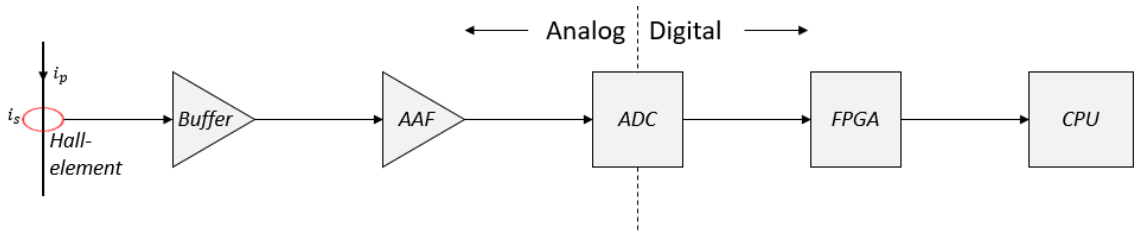


Figure 4.10: Current measurement circuit

The closed loop Hall effect sensor gives an exact representation of the input signal if the sensor is within its linear operation region. The instantaneous value of the secondary current of the sensor are given by $i_s = N_p/N_s \cdot i_p$, where N_p and N_s are the number of turns on the primary and secondary side of the senor, respectively. The secondary current passes through a measuring resistance producing a voltage drop over the resistor. The voltage over the resistor has the same shape as the input current and the scaling factor is given in the component's datasheet.

Output voltage of the Hall element is connected to a buffer circuit, e.g. differential amplifier. Scaling of the voltage signal according to the ADC can also be done in the buffer circuit. In order to avoid aliasing of the measurement, measuring signal is passed through an analog AAF (*anti-aliasing filter*). According to Nyquist-Shannon sampling theorem, an analog signal sampled at twice its frequency suffers form no loss of information [25]. In other words, the highest frequency that can be measured cannot exceed $f_{samp}/2$. The AFF can be realized by using different kinds of low pass filters. Ordinary first- and second-order LP filter or Butterworth filters are commonly used.

The ADC (*analog to digital converter*) converts the scaled and filtered analog signal to a digital number. Some of the most common converter techniques include Successive approximation, Single- and Dual slope integration, Flash and Delta-sigma ADC, all with their advantages and disadvantages [25].

The FPGA (*field programmable gate arrays*) can be used for several different tasks and is a critical component in the control of the machine. The FPGA used in the author's project consists of 125k programmable logical cells. These logical cells can be arranged and rearranged in order to complete the required tasks. The ability to arrange and rearrange connections between the logical cells, makes the FPGA able to isolate different sections of the logical structure to do different task in parallel. Going back the measuring circuit, the FPGA is used as a moving average filter. Since the sampling rate of FPGA is much higher than the CPU, filtering of the measured value from the ADC does not introduce an additional time delay in the calculations done by the CPU. The FPGA integrates the values from the ADC and the respective numbers of samples included in the integration within one sampling period of the CPU and stores the vales in the memory.

4.3.2 Offset error

When measuring the currents of a symmetrical three phase load with isolated neutral, it is common practice to measure two of the phases. The last phase current is easily calculated according to Kirchhoff's first law. If one of the measured phase voltages is subjected to a DC-offset, it can be understood that the average of the respective sinewave will no longer be zero. The DC-component introduced to the phase affected by the offset error, will also be present in the calculated phase current but with opposite sign. The consequences of current offset on the measured current vector $\underline{i}_s$ is best illustrated using a stator-oriented system.

Figure 4.11 shows measurements of $i_{s\alpha}^{s}$ and $i_{s\beta}^{s}$ as a function of time and $\underline{i}_s$ as a function of its respective α, β components. A DC-offset is imposed on one of the measured phases. The figure shows that the offset does not alter the amplitude of $i_{s\alpha}^{s}$ and $i_{s\beta}^{s}$, but introduced a DC-offset to one of the current components. By analysing $\underline{i}_s$ in the α^s, β^s-plane, it becomes evident that while keeping a circular trajectory, the offset causes the origin to move away from zero. As the origin of the circular trajectory drifts away from zero, amplitude of $\underline{i}_s$ starts to oscillate. The oscillations will have the same frequency as the electrical frequency in the system. In order to understand how this oscillation translates to a torque ripple, it is necessary to investigate the machine equations.

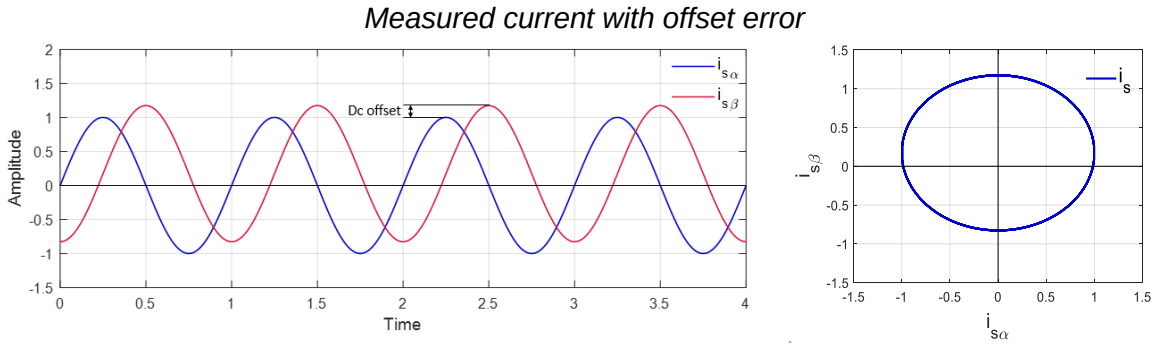


Figure 4.11: Offset error in current measurement

DC-offset in measurements is a consequence of unbalanced supply voltage to the current sensors or by one of the analog devices in the measuring circuit (e.g. buffer circuit, AFF or ADC). Since the offset is imposed by the measuring circuit itself, the measured currents can be expressed as:

$$\begin{aligned} i_{sa} &= i_{sa}^{*} + i'_{sa} \\ i_{sb} &= i_{sb}^{*} + i'_{sb} \\ i_{sc} &= -(i_{sa} + i_{sb}) \end{aligned} \tag{4.43}$$

Where: i_{sa}^{*} , i_{sb}^{*} are actual phase currents
i'_{sa} , i'_{sb} are DC-offsets

In order to more easily analyse the distorted current vector's effect on the rotor flux and electrical torque, the three-phase system is converted into a synchronous rotating reference frame. The decomposed, synchronous rotating current vectors are then given by:

$$\begin{aligned} i_{s\alpha}^{\psi_r} &= i_{s\alpha}^{\psi_r *} + i_{s\alpha}^{\psi_r \prime} \\ &= \frac{2}{3}\left[i_{sa} \cdot \cos(2\pi f_s) + i_{sb} \cdot \cos\left(\omega_s - \frac{2\pi}{3}\right) + (-i_{sa} - i_{sb}) \cdot \cos\left(\omega_s + \frac{2\pi}{3}\right)\right] \end{aligned} \tag{4.44}$$

$$\begin{aligned} i_{s\beta}^{\psi_r} &= i_{s\beta}^{\psi_r *} + i_{s\beta}^{\psi_r \prime} \\ &= \frac{2}{3}\left[-i_{sa} \cdot \sin(2\pi f_s) - i_{sb} \cdot \sin\left(\omega_s - \frac{2\pi}{3}\right) - (-i_{sa} - i_{sb}) \cdot \sin\left(\omega_s + \frac{2\pi}{3}\right)\right] \end{aligned} \tag{4.45}$$

As with the three-phase system, $i_{s\alpha}^{\psi_r}$ and $i_{s\beta}^{\psi_r}$ are expressed as an actual current vector and a DC-offset contribution. However, due to the transformation, $i_{s\alpha}^{\psi_r{}^*}$ and $i_{s\beta}^{\psi_r{}^*}$ becomes DC-quantities and $i_{s\alpha}^{\psi_r{}'}$ and $i_{s\beta}^{\psi_r{}'}$ becomes sinusoidal.

If the current measurement is used in a feedback loop of a quick current controller, oscillation in the measured signal will be compensated for by the controller itself. This is done by introducing an oscillatory component in the controller output. The oscillatory component is the same error as previously discussed, but instead of being present in current measurements, it now creates an oscillation in the actual phase current. The phase current can therefore be expressed in terms of a constant reference value (in steady state) and an oscillating error.

$$i_{s\alpha}^{\psi_r{}^*} = i_{s\alpha}^{\psi_r} + i_{s\alpha}^{\psi_r{}'} = i_{s\alpha,ref}^{\psi_r} + i_{s\alpha}^{\psi_r{}'} \tag{4.46}$$

$$i_{s\beta}^{\psi_r{}^*} = i_{s\beta}^{\psi_r} + i_{s\beta}^{\psi_r{}'} = i_{s\beta,ref}^{\psi_r} + i_{s\beta}^{\psi_r{}'} \tag{4.47}$$

The rotor flux of the machine is given by eq. (4.48). The rotor time constant, T_r, is large compared to other time constants in the system [11]. Due to the large time constant it can be understood that fast oscillation in $i_{s\alpha}^{\psi_r{}^*}$ has little impact on the amplitude of ψ_R. It can therefore be concluded that the offset current does not influence ψ_R and the flux can be assumed constant [18].

$$\frac{d\psi_R}{dt} = -\frac{1}{T_r}\cdot\psi_R + \frac{x_M}{T_r}\cdot i_{s\alpha}^{\psi_r} \tag{4.48}$$

The torque equation in pu is presented in eq. (4.49). By inserting eq. (4.47), the electrical torque can be represented by a reference torque and a disturbance load. Oscillations in $i_{s\beta}^{\psi_r{}^*}$, clearly causes oscillation in the electrical torque generated by the machine. As stated earlier, oscillation in the torque can be represented as a disturbance load torque.

$$\tau_e = \psi_R \cdot i_{s\beta}^{\psi_r{}^*} = \psi_R \cdot \left(i_{s\beta,ref}^{\psi_r} + i_{s\beta}^{\psi_r{}'}\right) = \tau_{e,ref} - \Delta\tau_L \tag{4.49}$$

The oscillation in the torque is given by the frequency of $i_{s\beta}^{\psi_r{}'}$ in the synchronous rotating reference system. It is given in [18] that $i_{s\beta}^{\psi_r{}'}$ can be expressed as:

$$\begin{aligned} i_{s\beta}^{\psi_r{}'} &= \frac{2}{3}\left[-\sin(\omega_s)\cdot i'_{sa} - \sin\left(\omega_s - \frac{2\pi}{3}\right)\cdot i'_{sb} - \sin\left(\omega_s + \frac{2\pi}{3}\right)\cdot(-i'_{sa} - i'_{sb})\right] \\ &= \frac{2}{\sqrt{3}}\sqrt{(i'_{as})^2 + (i'_{as}\cdot i'_{bs}) + (i'_{bs})^2}\cdot\cos(\omega_s + \gamma) \end{aligned} \tag{4.50}$$

Where: $\gamma = \arctan\left(\frac{\sqrt{3}\cdot i'_{as}}{i'_{as}+2i'_{bs}}\right)$

The disturbance load torque can then be expressed as follows:

$$\Delta\tau_L = \psi_R\frac{2}{\sqrt{3}}\sqrt{(i'_{as})^2 + (i'_{as}\cdot i'_{bs}) + (i'_{bs})^2}\cdot\cos(\omega_s + \gamma) \tag{4.51}$$

Eq. (4.51) shows that DC-offset in the current measurement causes a disturbance load that oscillates equal to the stator frequency, thus causing vibrations in the machine with equal frequency.

4.3.3 Scaling error

The other current measurement fault that causes torque ripples is called scaling error. As mentioned in section 4.3.1, measuring signal from the current sensor has to be scaled according to the input of the ADC. The resistance used in the current measurement itself and in the scaling circuit, contain some degree of inaccuracy. This inaccuracy may lead to a misrepresentation of amplitude when the current controller rescales the signal back to the real current values. A scaling error in one of the two measured phases will also affect the amplitude of the calculated phase current. As for the last sub-chapter, the scaling errors effect on measured current is best illustrated in stator-oriented reference frame.

Figure 4.12 shows measurements of $i_{s\alpha}^{s}$ and $i_{s\beta}^{s}$ as a function of time and $\underline{i}_s$ as a function of its respective α, β components. The measuring circuit is imposed with a scaling error in one of the measured phase-currents. It can be seen from the figure that this particular scaling error effects the amplitude of $i_{s\alpha}^{s}$. Looking at $\underline{i}_s$ in the α^s, β^s-plane shows that the scaling error changes current vectors trajectory from a circular to an elliptic trajectory. The elliptic trajectory causes amplitude of $\underline{i}_s$ to oscillate with twice the electrical frequency.

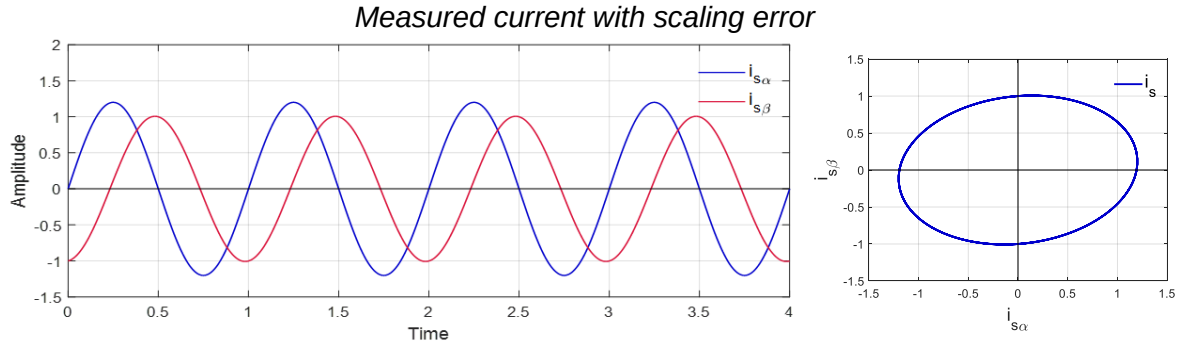


Figure 4.12: Scaling error in current measurement

If the current measurement is used in a feedback loop of a fast current-controller, it can be assumed that the measured value follows the given reference value. This assumption is valid because the current controller is much quicker than the changes in the reference value. Instantaneous value of the measured current can be expressed as:

$$i_{sa} = i_{sa,ref} = \hat{I}\cos(\omega_s) \tag{4.52}$$

$$i_{sb} = i_{sb,ref} = \hat{I}\cos\left(\omega_s - \frac{2}{3}\pi\right) \tag{4.53}$$

Since the measured value follows the reference value, it can be understood that an error in the measured signal will lead to the actual current being regulated incorrectly. As an example, if measured current indicates an amplitude which is 2% higher than the actual current, output current will be regulated to a value lower than the reference value. Given that the actual currents are controlled by a high-performance current controller, they can be expressed as:

$$i_{sa}^{*} = \hat{I}\cos(\omega_s)/K_a \tag{4.54}$$

$$i_{sb}^{*} = \hat{I}\cos\left(\omega_s - \frac{2}{3}\pi\right)/K_b \tag{4.55}$$

Where K_a and K_b are the scaling error introduced by the measuring circuit

It was explained in section 4.3.2 that oscillations in $i_{s\alpha}^{\psi_r{}^*}$ due to a DC-offset had little effect on the rotor flux amplitude due to the large rotor time constant. This is also the case when the measurements are subjected to scaling errors. It was also shown that $i_{s\beta}^{\psi_r{}'}$ created an oscillatory disturbance torque. The same analysis can be conducted for measurements subjected to scaling errors. $i_{s\beta}^{\psi_r{}'}$ can be found by preforming a Park transformation on the three-phase system.

$$i_{s\beta}^{\psi_r{}'} = \frac{2}{3}\left[-\sin(\omega_s)\cdot i'_{sa} - \sin\left(\omega_s - \frac{2\pi}{3}\right)\cdot i'_{sb} - \sin\left(\omega_s + \frac{2\pi}{3}\right)\cdot i'_{sc}\right] \tag{4.56}$$

i'_{sa} and i'_{sb} are the difference between the measured current and the actual current. Using equations (4.52)-(4.55), the error between measured and actual current can be expressed as:

$$i'_{sa} = i_{sa} - i^*_{sa} = \hat{I}\cos(\omega_s)\left(1 - \frac{1}{K_a}\right) \tag{4.57}$$

$$i'_{sb} = i_{sb} - i^*_{sb} = \hat{I}\cos\left(\omega_s - \frac{2}{3}\pi\right)\left(1 - \frac{1}{K_b}\right) \tag{4.58}$$

$$i'_{sc} = -(i'_{sa} + i'_{sb}) \tag{4.59}$$

The expression for $i_{s\beta}^{\psi_r{}'}$ can be further developed by inserting equations (4.57)-(4.59) in eq. (4.56). With use of Maplesoft™ mathematical program Maple and some geometric simplifications, $i_{s\beta}^{\psi_r{}'}$ can be expressed as:

$$\begin{aligned} i_{s\beta}^{\psi_r{}'} &= \frac{K_a - K_b}{K_a\cdot K_b}\cdot\frac{\hat{I}}{3}\cdot\left(\sqrt{3}\cos(\omega_s) - 3\sin(\omega_s)\right)\cdot\cos(\omega_s) \\ &= \frac{K_a - K_b}{K_a\cdot K_b}\cdot\frac{\hat{I}}{6}\cdot\left(2\sqrt{3}\sin\left(\frac{\pi}{6} - 2\omega_s\right) + \sqrt{3}\right) \\ &= \frac{K_a - K_b}{K_a\cdot K_b}\cdot\frac{\hat{I}}{\sqrt{3}}\cdot\left(\sin\left(\frac{\pi}{6} - 2\omega_s\right) + \frac{1}{2}\right) \\ &= \frac{K_a - K_b}{K_a\cdot K_b}\cdot\frac{\hat{I}}{\sqrt{3}}\cdot\left(\cos\left(2\omega_s + \frac{\pi}{3}\right) + \frac{1}{2}\right) \end{aligned} \tag{4.60}$$

Eq. (4.60) shows that $i_{s\beta}^{\psi_r{}'}$ oscillates with double the stator frequency. Going back torque equation shown in (4.49), it is possible to find an expression for the disturbance load $\Delta\tau_L$.

$$\begin{aligned} \Delta\tau_L &= \psi_R\cdot i_{s\beta}^{\psi_r{}'} \\ &= \frac{K_a - K_b}{K_a\cdot K_b}\cdot\frac{\hat{I}\cdot\psi_R}{\sqrt{3}}\cdot\left(\cos\left(2\omega_s + \frac{\pi}{3}\right)\right) + \frac{K_a - K_b}{K_a\cdot K_b}\cdot\frac{\hat{I}\cdot\psi_R}{2\sqrt{3}} \end{aligned} \tag{4.61}$$

Eq. (4.61) shows that a scaling error in a measured phase-current causes a disturbance load that oscillates with twice the stator frequency. As in previous section, the variation in electrical torque causes vibrations in the machine equal to twice the stator frequency.

5 Programming Structure and Software

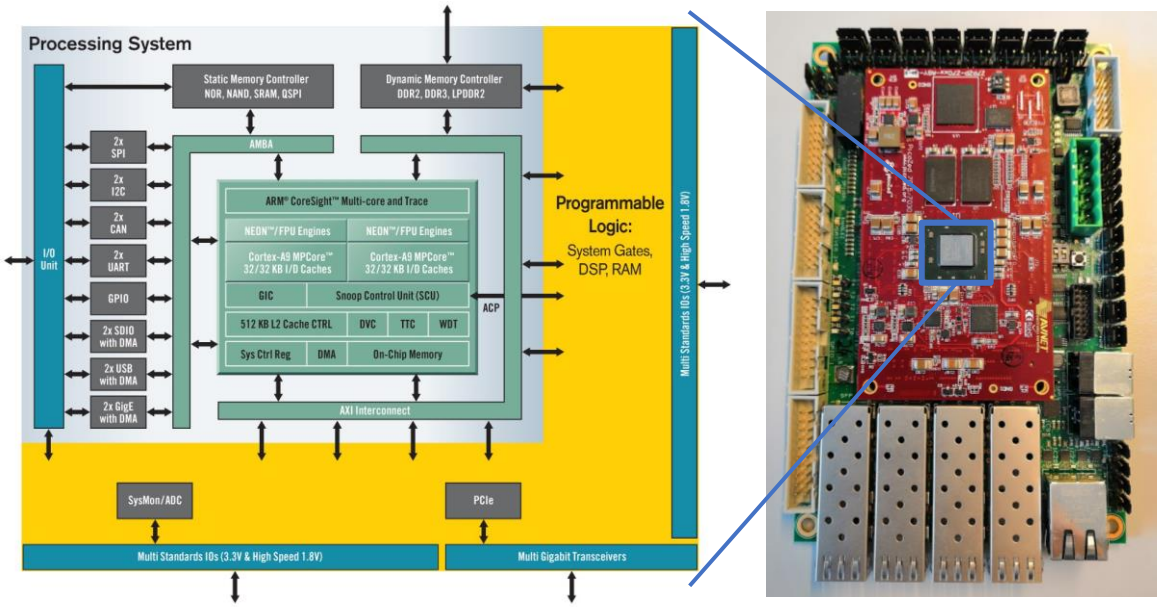
5.1 Introduction

An R&D group called PESC at the department of Electrical Power Engineering has since the beginning of 2019 worked on establishing a common NTNU Control Platform. The control platform is intended to be a common foundation for research projects and a starting point for future master and doctor thesis at the department. Contributing to the PESC group has been a major part of the author's master thesis. The control platform is built by writing new and modifying already existing C++ code. Also, some of the IP-cores used in FPGA programming are provided by SINTEF

This chapter gives a quick overview of the controller boards and Software on which the NTNU Control Platform is based upon. Programming structure of the CPU and an introduction to Xilinx System Generator tool for creation of IP-cores are also presented in this chapter.

The control platform uses a control board from Avnet called PicoZed 7030, red board in Figure 5.1-(b), and a process interface board from SINTEF Energy, green board in Figure 5.1-(b). The control board from Avnet uses a Zynq xc 7z030 processor that has two floating point ARM-processors (also called CPUs) and one FPGA mounted on the same chip. The processors physical location on the board and an illustration of the Zynq-processor is also shown in Figure 5.1. One of the CPU runs a Linux program that is used to program the remaining CPU and FPGA. It also allows the card to be programmed and monitored using special release and monitoring tools. Linux program, release tools and monitoring tools are provided by The Switch Marine Drives. Process interface board from SINTEF allows for control of two 2-level 3-phase inverters and eight measurements (normally $2x$3-phase current measurement and $2x$DC-Link voltage measurement).

Programming of control board is done using three different software, all provided by Xilinx. Code for the remaining CPU is written in C++ using Software Development Kit (SDK). FPGA is programmed by connecting so-called IP-cores together in a program called Vivado. The IP-cores them self are made using conventional VHDL-Programming or by using Xilinx's System Generator plugin in Simulink. Vivado also provide a generic IP-core catalogue that can be used.



(a) (b)

Figure 5.1: (a) schematic representation of the Zynq-processor [26]. *(b) Photo of the control boards.*

5.2 Programming structure in SDK

The CPU program is structured in such a way that it is possible to change the application with minimum modifications to the code. The core structure consists of a "main.cc" file that runs in between the interrupts and a "DriveRoutioneFast.cpp" file that calls all the functions of the drive. In the future, the core structure will also contain a State Machine located in the DriveRoutineFast, based on the Drivecom standard. The objects that is used within this foundation are made using class declaration and can be easily changed or added to a code repository for future use. In this subchapter main.cc and DriveRoutineFast.cpp will be presented and explained. The chapter will also give an example of class declaration.

A simplified version of "main.cc" is presented in Code 5.1. The "main.cc" file stats with initializing all the required communication, parameters and interrupts before entering an infinite while loop. The program stays in the while loop until the CPU receives an interrupt generated by the PWM IP-core. When the CPU receives the interrupt, the program jumps out of the while loop and executes "void PWMIsr()", shown between line 1-6 in the code. The function runs DriveRoutineFast, thus executing one iteration of the control routine. It is important to note that DriveRoutineFast needs to have sufficient with time to finish all its calculations before a new interrupt appears. However, today's fast processing capability reduces this problem. In fact, the processors used on the control board is fast enough to use only one interrupt for the entire drive control if desired.

```
1. void PWMIsr()            // Interrupt from the PWM
2. {
3.   DriveRoutineFast();
4.
5.   m_dataloggers[0].Run(DATALOGGER_PROBE_BACKGROUND);
6. }
7.
8. int main()
9. {
10.  // Initialize SW and parameters from data base:
11.  Initialize_SW();
12.
13.  // Initialize Interrupts
14.  // Initialize Communication
15.
16.
17.  while(true)
18.  {
19.          ++cntr_background;
20.          TESTLED_REG =light;
21.          g_wtcp.Service();
22.  }
23.  return 0;
24.  }
```

Code 5.1: main.cc

A summary of DriveRoutineFast.cpp is presented in Code 5.2. The code shows the different objects that are executed when the CPU receives an interrupt. These can be seen between line 17-35. Line 6-15 shows the global variables used to operate the drive. The global variables include both input variables like FrequencyIn and output variables like u_a_est. Input and output variables are monitored and changed using the Software "WatchView", developed and provided by The Switch. Line 1-4 shows the class declarations. Code for the classes are written in individual .cpp files with a corresponding heather file ".hpp". Since type of converter and control strategy etc. is specified through class declaration, one can simply change the control strategy from V/f-control to FOC by including a

new heather file and changing the name of the class. This object-oriented programming makes the code very flexible and versatile.

```
1. //Class declarations
2.   PWM_2L_3ph              PWM_1;
3.   ScalarControl           Controller;
4.   VoltageEstimator        BlankingTimeCorr;
5.
6. //Global variables
7.   int FrequencyIn;        //Desired frequency
8.   float BlankingTime;     //Blanking time in [s]
9.   float DCLinkVoltage;
10.  float U_s_amp;          //Calculated controller volt.vect.amp
11.  float U_s_ang;          //Calculated controller volt.vect.ang
12.  float u_a_est;          //Estimated a-phase voltage
13.  float u_b_est;          //Estimated b-phase voltage
14.  float u_c_est;          //Estimated c-phase voltage
15.  int NonLinCorr;         //0=off, 1=on with non-linear corr.
16.
17.   void DriveRoutineFast(void)
18.  {
19.          //Voltage measurement estimation
20.          BlankingTimeCorr.Run(BlankingTime);
21.
22.          u_a_est = BlankingTimeCorr.GetPhaseVoltage_a();
23.          u_b_est = BlankingTimeCorr.GetPhaseVoltage_b();
24.          u_c_est = BlankingTimeCorr.GetPhaseVoltage_c();
25.
26.  //Runs the converter
27.  Controller.Run(FrequencyIn,5);
28.
29.  //Return voltage vector
30.  U_s_amp = Controller.GetVoltageVectorAmp();
31.  U_s_ang = Controller.GetVoltageVectorAng();
32.
33.  //Runs PWM
34.  PWM_1.Run_3phase(U_s_amp, U_s_ang, DCLinkVoltage, NonLinCorr);
35.  }
```

Code 5.2: DriveRoutineFast.cpp

A common programming platform with multiple authors has the potential to generate programs with different “programming styles”, making it over-complex and hard to understand. It is therefore necessary to create a common structure, describing how future classes should be written. A suggestion of such a structure is presented in Code 5.3 and Code 5.4, showing the heather and source file of an example-class, respectively.

The first line in the .hpp file presented in Code 5.3 defines the name of the class. The class name is used to create objects in for example DriveRoutineFast.cpp, as shown in line 2-4 in Code 5.2. It is possible to create several objects of one class. This can be useful when e.g. two identical inverters are controlled by the same processor. The class is built up by public and private functions and values. It is good practice to keep all variables within a class private (line 9-19) and functions that sets, gets or calculate the private variables as public (line 3-7).

```
1. class ExampleClass
2.   {
3. public:
4.   void Init (int InitValueIn_1, int InitValueIn_2);
5.   void Run (int VariableIn_1, float VariableIn_2);
6.   void SetValue (float ParameterIn_1);
7.   float GetValue (void);
8.
9. private:
10.          // Initialized values
11.          int InitValue_1;
12.          int InitValue_2;
13.
14.          // Calculated values
15.          float temp_1;
16.          float temp_2;
17.
18.          // Output values
19.          float Output;
20.  };
```

Code 5.3: ExampleClass.hpp

Source file of the example-class specifies the actions that will happen when the function is called up in DriveRoutineFast or other places within the program. The example code shows how variables can be set during the initialisation run (line 3-7), how input variables can be used in combination with private variables to execute a calculation (line 9-14) and how a private variable can be accessed outside the class (line 16-19).

```
1. #include "ExampleClass.hpp"
2.
3. void ExampleClass::Init(int InitValueIn_1, int InitValueIn_2)
4.          {
5.          InitValue_1 = InitValueIn_1;
6.          InitValue_2 = InitValueIn_2;
7.          }
8.
9. void ExampleClass::Run(int VariableIn_1, float VariableIn_2)
10.         {
11.         temp_1 = InitValue_1 + VariableIn_1;
12.         temp_2 = InitValue_2 + VariableIn_2;
13.         Output = temp_1 * temp_2;
14.         }
15.
16.  float ExampleClass::GetValue (void)
17.         {
18.         return Output;
19.         }
```

Code 5.4: ExampleClass.cpp

5.3 IP-core Generation using System Generator DSP in Simulink

As a part of the installation of Vivado, Xilinx also provide the option to install a tool called System Generator DSP. The System Generator tool becomes an extension to MATLAB Simulink and allows for programming and generation of IP-cores using Simulink. In order to generate an IP-core, special Simulink blocks provided by Xilinx have to be used. The new library gets implemented as a part of the installation. Once a block design is complete, it is possible to test the functionality using other Simulink libraries.

Gateway In and Gateway Out

The interface of the new IP-core is defined using the blocks *Gateway In* and *Gateway Out*. This is shown in Figure 5.2 where the violet area represents the inside of the IP-core. The two gateway ports convert signals from Simulink, FPGA or CPU to fixed- or floating-point signal or vice versa. Output type from Gateway In can be specified using the block settings highlighted in Figure 5.3-(a). The Gateway can convert input data to Boolean (true, false), Fixed-point (signed or unsigned) or floating-point (single or double). Figure 5.3-(b) shows gateway implementation in the IP-core design and are the same for both Gateway In and Gateway Out. Gateway implementation defines how the software communicates with the IP-core. The specification "None" is used if the input or output is connected to another IP-core and "AXI4-Lite" is used for communication with the CPU. The new IP-core can also generate its own interrupt that can be used to trigger special events in the CPU.

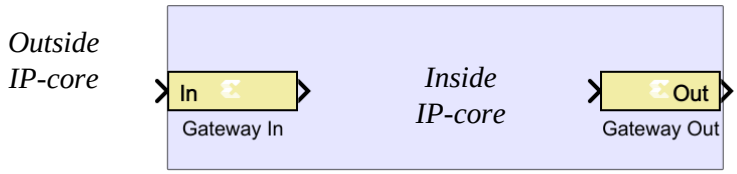
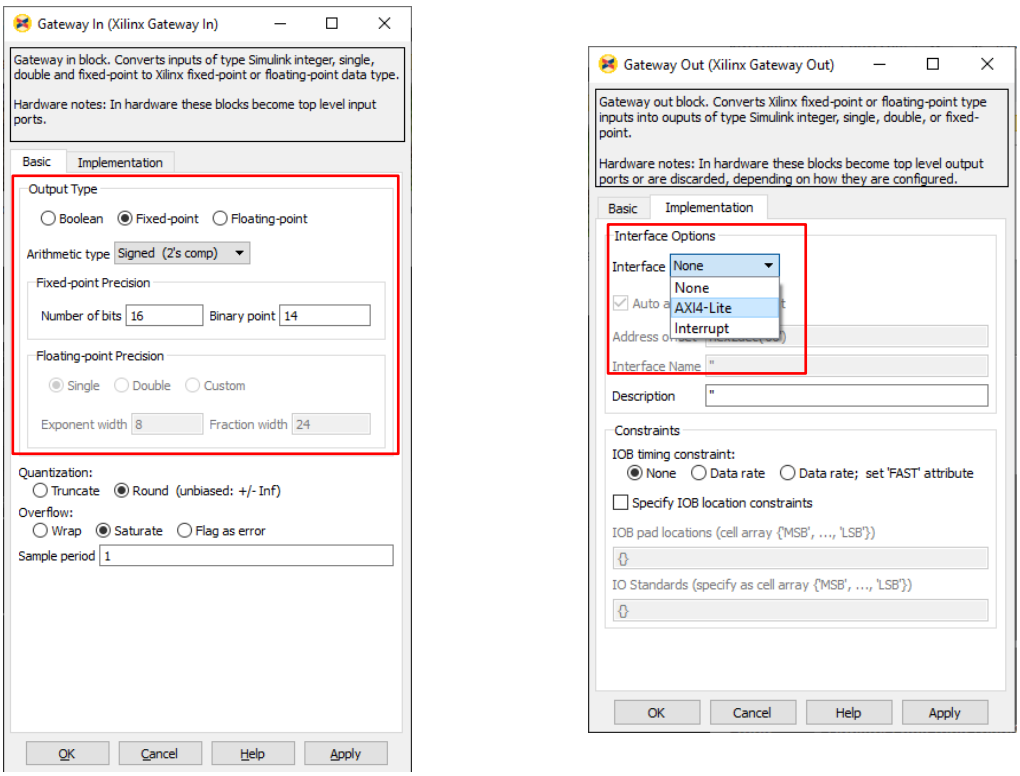


Figure 5.2: Input and Output gates



(a) (b)

Figure 5.3: Block specification for Gateway ports

Creating a block design

As mentioned earlier, when downloading System Generator, Xilinx will provide a Simulink library dedicated to generating IP-cores. The new blocks provided by Xilinx can be used in a similar manner as other Simulink libraries, making programming easier and more understandable for many electrical engineers without special knowledge of VHDL programming. The fact that blocks are made by experts from Xilinx ensures a reliable IP-cores design.

A picture of the basic building blocks is shown in Figure 5.4. Enable ports and reset ports can also be added to many of the blocks, making them very versatile. It is also possible to integrate already existing MATLAB scrips (.m files) and VHDL files in the new design using the block “Black Box”.

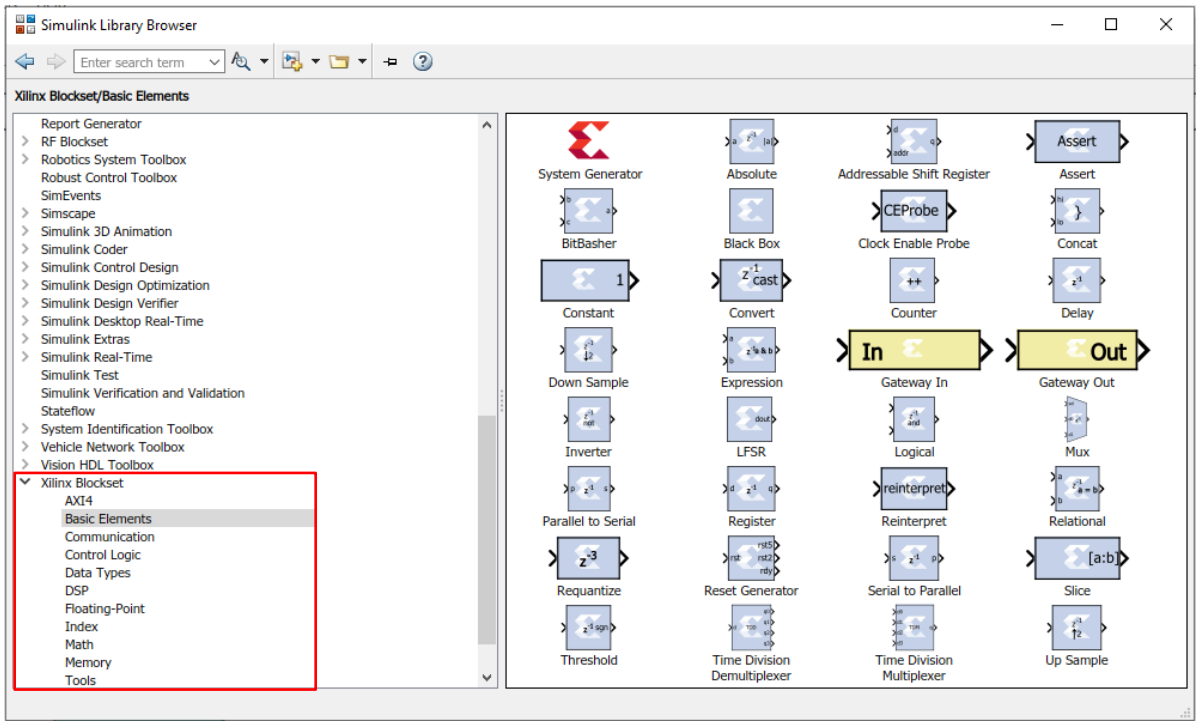


Figure 5.4: Basic block library for the Xilinx tool

An example of a block design is shown in Figure 5.5. The design is created using a counter-, relational- and constant-block and generates a sawtooth signal with the amplitude of 100 at the output as long as the counter is enabled.

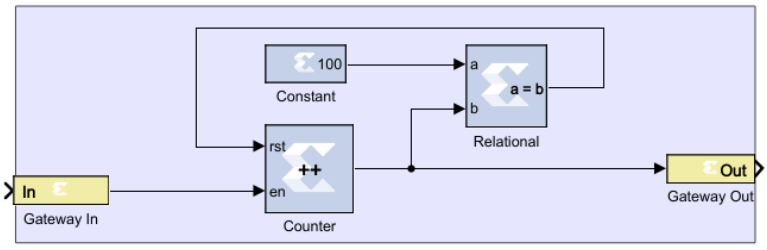


Figure 5.5: Counter example

Simulating the block design

The gateway ports allow the IP-core to be simulated before it is implemented it in the FPGA design. Simulation is a quick way of verifying the design. The input signal to the IP-core can be anything from a simple constant or a sinewave generator to the output of another Simulink model. The output of the IP-core can be monitored or used in another IP-core design. A simulation of the counter constructed above can be performed using a step function. The complete simulation circuit is presented in Figure 5.6. Data type of the input signal can be specified using a *Data Type Conversion* block and the converted signal is then passed into the IP-core.

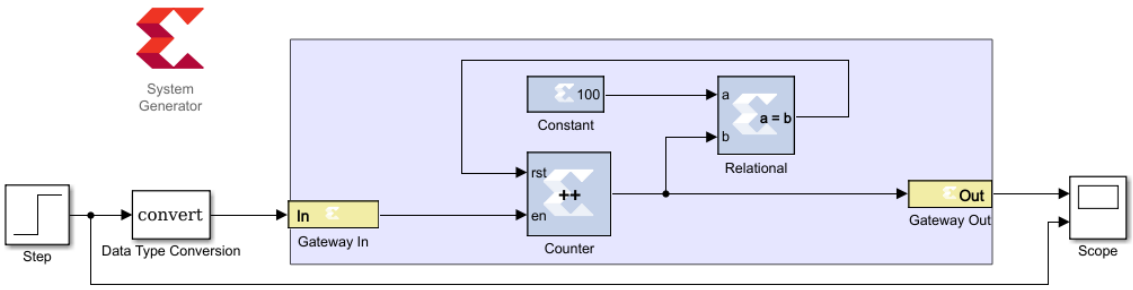


Figure 5.6: Simulation of counter

Simulation results can be monitored using a scope as shown in Figure 5.6. Scope output is presented in Figure 5.7. The counter counts to 100 where it is reset. Also, the counter does not start until it is enabled by the step function. In order to run simulations, one has to specify a sampling time. The sampling time is given by the internal clocks of the FPGA and can be found in Vivado. This simulation is executed with a sampling time of 10 μs. It is also necessary to include System Generator block in the Simulink design to get the program to compile.

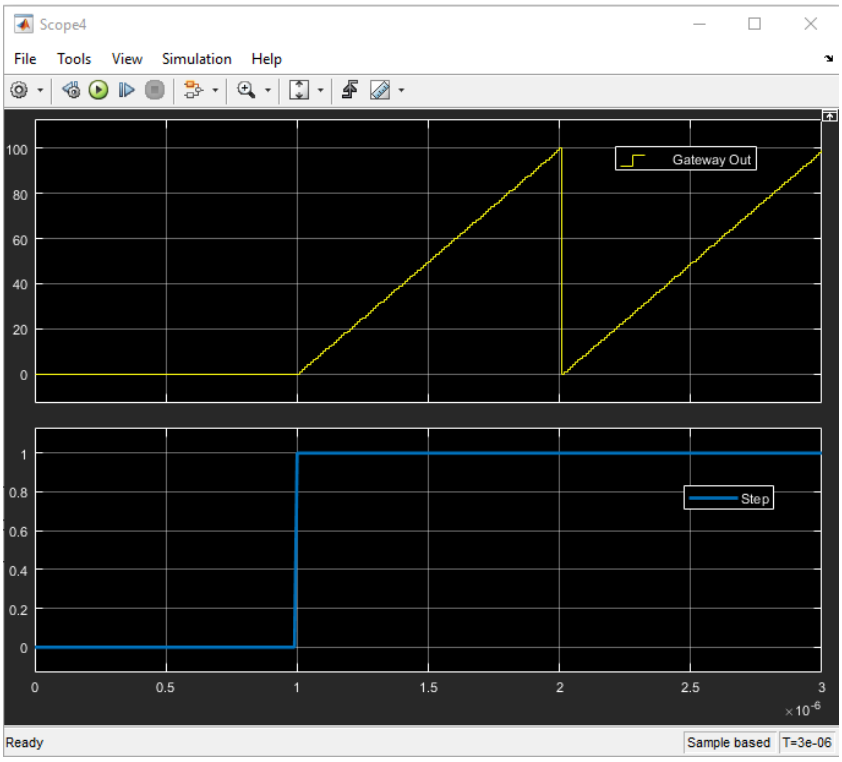
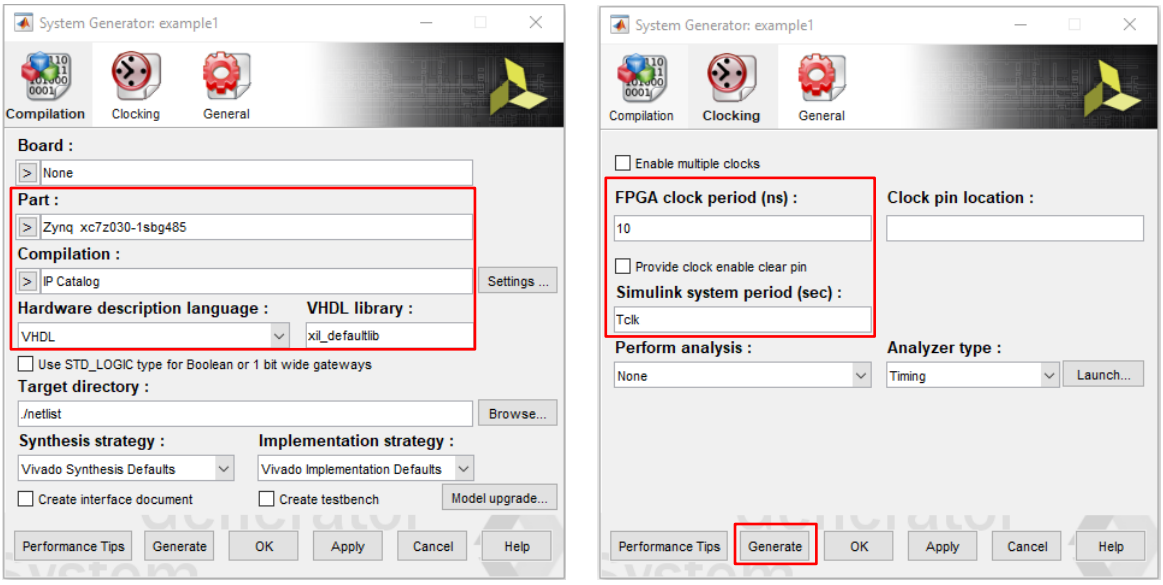


Figure 5.7: Scope output

Generating an IP-core

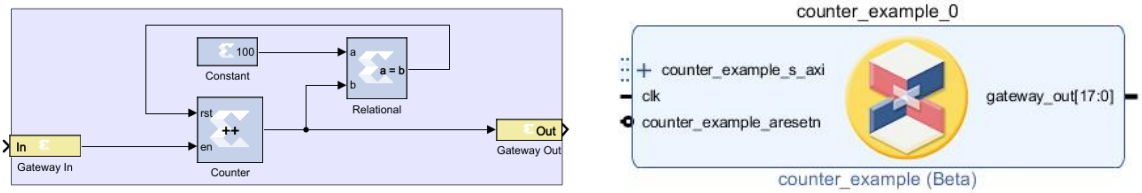
When the program is sufficiently tested it is possible to generate an IP-core that can be integrated in the FPGA design. This is done by double clicking on the System Generator block shown on the top left corner of Figure 5.6. The options menu is shown in Figure 5.8. Here, one has to specify which board or processor that is being used and what hardware language the code should be generated in. These options are highlighted in Figure 5.8-(a). Next, the clock used in FPGA and in Simulink simulations are specified. If the clocks do not match up, the system generator will take care of the scaling. The clock specification is found under the banner "Clocking" and is highlighted in Figure 5.8-(b). Once the settings are properly chosen, the IP-core is created by pressing "Generate".



(a) (b)

Figure 5.8: System Generator setup

Figure 5.9-(a) shows the counter circuit from Simulink and Figure 5.9-(b) shows the final generated IP-core that can be implemented in the FPGA design. In this specific design, Gateway In was specified as AXI4-Lite and Gateway Out was specified as None. Enable of the counter is controlled from the CPU through the AXI-port of the block. The AXI-port is the top left input port of Figure 5.9-(b). This connection will be auto-generated when the new IP-core is added to the FPGA design, together with the clock input "clk". The output port of the design, Gateway Out, is shown as an internal connection. This connection has to manually be connected the desired IP-core and will not be accessed from the CPU.



(a) (b)

Figure 5.9: (a) Counter example in Simulink. (b) Counter example in Vivado

Implementing the IP-core in SDK

Once the FPGA program in Vivado is completed and a new bit stream has been created, it is possible to export a new hardware platform. The new hardware platform contains a special made .h file for the new IP-cores generated by System Generator. The .h file allow for easy read and write access for the CPU. A very nice feature of this system is that the user does not need to know about the memory location for the different in- and outputs of the IP-core. The communication between CPU and IP-core is established by initializing the IP-core with a standardized syntax and the in- and outputs of the IP-core are written to or read from using pre-defined function. An example of how to initialize, write to and read from an IP-core is presented below.

The hardware platform generated from Vivado is linked to SDK and automatically updated if changed. Each project inside SDK is also linked to a Board Support Package (BSP) which again is liked to a hardware platform. The links between hardware platform, BSP and SDK project is shown in Figure 5.10. If the hardware platform is updated, SDK will ask if the user want to update the BSP. Since the project uses .h files directly from the BSP, all driver in the project is also updated once the BSP is updated

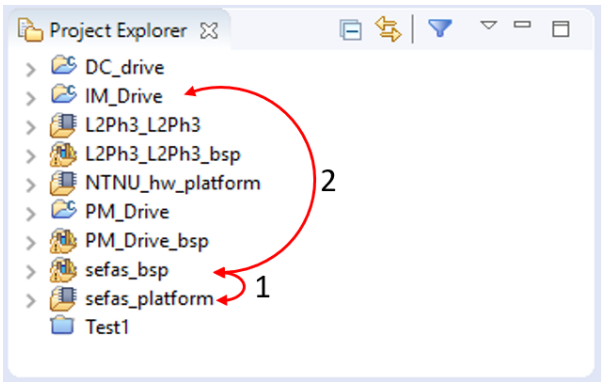


Figure 5.10: Link between hardware platform, BSP and project

When the BSP has been updated, the auto-generated .h file can be included in the initialization file of the project. The .h file name is generated using the same name as shown in the Vivado design. From Figure 5.9-(b) it is seen that the block is labelled “counter_example_0” where “counter_example” is the name of the block and “0” is the block ID. It can be understood that if a second Counter IP-core was to be added to the FPGA design, it would be named “counter_example_1”. Initialization of the counter IP-core is shown in Code 5.5. The first step of initializing the new IP-core is to create a pointer for that specific IP-core. This is shown in line 5. The last step is to run the initialize function specified in the .h file of the IP-core. The initialize function is shown in line 10 of the code. The initialize function can be generalized to “Name”_Initialize(&pointer, ”block ID”). In this case, memory location of “ptr_counter” and the block ID “0” is passed to the function.

```
1. // Include auto-generated .h file from BSP
2. #include "counter_example.h"
3.
4. //User IP-core variables
5.  counter_example         ptr_counter;
6.
7. void Initialize (void)
8.  {
9.   //Initialize the new IP-core
10.  counter_example_Initialize(&ptr_counter,0 //Insert block ID);
11.  }
```

Code 5.5: Initialization of a new IP-core

After the IP-core has been initialized, input and output values from the IP-core can be easily accessed by standardized functions. A generalized syntax for the read and write functions are presented below:

- "Name"_"Port name"_read(&pointer);
- "Name"_"Port name"_write(&pointer, "value");

The name of the IP-core is the same as during initialization. Port name is the same as the Gateway port in Simulink. In the Counter example, this name will be "gateway_in". Block name, port names and the functions presented can also be found in the block datasheet located in the hardware platform.

An example of how to read and write to an IP-core is presented in Code 5.6. Line 13 shows how "gateway_in" can be assigned the value 1, thus enabling the counter. The input status can be monitored using the read function shown in line 16. It is important to note that before the read and write functions can be accessed in a source file, .h for the IP-core in question has to be included, shown in line 2. Moreover, the pointer also has to be declared in similar manner as in the initialization file. This is done by using an extern declaration and is shown in line 5 of the example code.

```
1. // Include auto-generated .h file from BSP
2. #include "counter_example.h"
3.
4. // Looks for external declaration in "Counter_example.h"
5. extern counter_example   ptr_counter;
6.
7. void RunRoutine (void)
8.   {
9.   int EnableValue;
10.
11.  //********Communication to the IP-core********//
12.  //Write to the IP-core
13.  counter_example_gateway_in_write(&ptr_counter,1);
14.
15.  //Read from the IP-core
16.  EnableValue = counter_example_gateway_in_read(&ptr_counter);
17.  }
```

Code 5.6: Reading and writing to and from the IP-core

6 CPU and FPGA Programming

6.1 Introduction

This chapter will show how the programming tools presented in chapter 5 can be used to create an open loop V/f-control of a three-phase induction machine. It will also show how the phase voltage can be estimated using IGBTs control signal with correction for banking time. Lastly, implementation of a correction voltage to the control voltage will be presented.

The program is built on the control platform developed in the PESC group. A summary of main.cc and DriveRoutineFast.cpp is shown in Code 5.1 and Code 5.2, respectively. The program is built using different objects in the CPU and different IP-cores in the FPGA. These elements are illustrated in Figure 6.1. The project specific classes developed in this thesis are: V/f-controller, PWM modulator and voltage estimator and corrector. The classes have been written in C++ using SDK and implemented in the standardized control platform. The PWM IP-core is developed by SINTEF and the Blanking Time Correction IP-core is developed by the author using Xilinx's System Generator tool in Simulink.

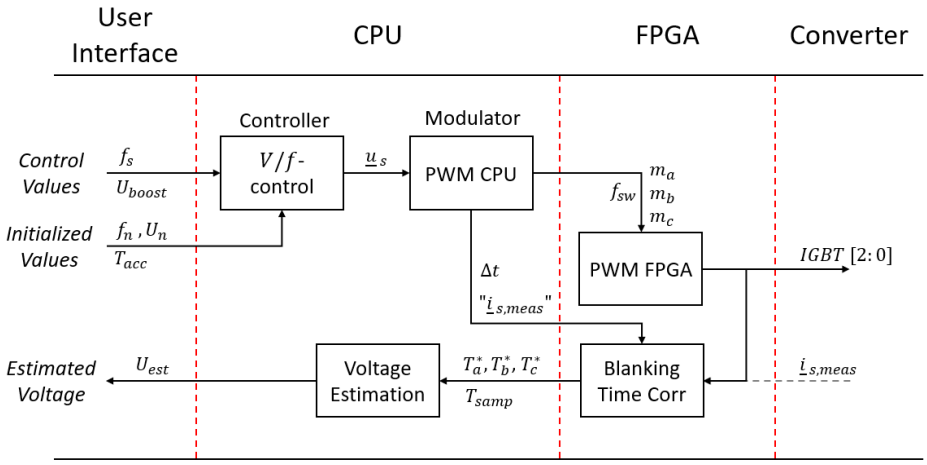


Figure 6.1: Objects and elements in the control software

The V/f-controller gathers information about the machine's nominal frequency and voltage together with desired acceleration time (from 0-1 pu speed) during the initialization run. Control values f_s and U_{boost} can be changed during operation and controls stator frequency and voltage boost at low speed. The output of the controller is a voltage vector amplitude and angle. Since both open loop V/f-control and closed loop FOC has the same output, control strategy can easily be changed without having to major modification to the program.

The PWM receives a voltage vector from the controller and decompose it into α^s, β^s components. Then an inverse Clark-Transformation is performed in order to find the three phase voltages. The control voltage, or modulation index, and switching frequency are sent to the PWM IP-core.

Blanking Time Correction IP-core uses the three bit-signals for T_{A+}, T_{B+}, T_{C+} (see Figure 4.1) from PWM IP-core and blanking time duration, Δt, as inputs. Current measurement, $\underline{i}_{s,meas}$, should have come from Hall elements connected to a running machine. However, since all testing in this thesis is performed without a physical inverter, a current signal is emulated in the modulator and sent to the FPGA. The IP-core gives out the number of samples where bridge-leg midpoint has the voltage potential of U_{dc} within one interrupt. It also sends out the total number of samples within one interrupt.

Voltage estimator uses the numbers of samples and DC-link voltage to estimate phase voltages.

6.2 V/f- Control (Controller)

The V/f-controller was presented in chapter 3.1. The program is based on an open loop control with a boost voltage in the lower speed range. Boost voltage is introduced to compensate for the resistive voltage drop. The program is also designed to go into field weakening when the frequency becomes higher than nominal frequency. Voltage/frequency curve of the controller is illustrated in Figure 6.2. The figure shows a linear relation between voltage and frequency with a voltage lift in the lower speed regions, and constant voltage with increased frequency in the field weakening range.

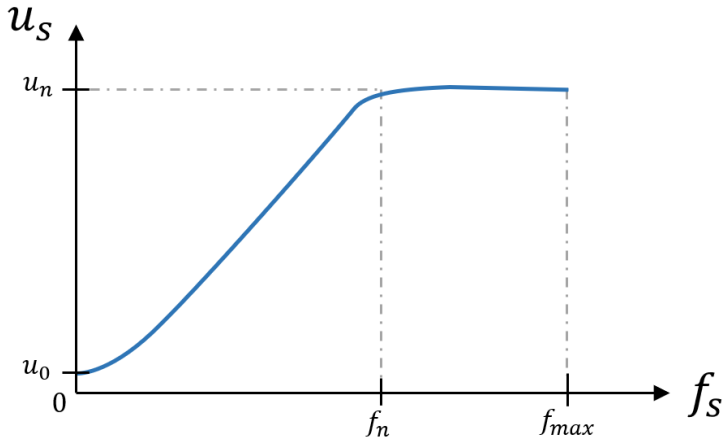


Figure 6.2: V/f-curve

Initialization of the controller is shown Code 6.1. The controller is assigned a nominal voltage and frequency, switching frequency and desired acceleration time. The values are passed to the private variables of the class, so they can be accessed by any member of the class. Lastly, peak value of the phase voltage is found, as shown in line 10.

```
1. void ScalarControl::Init(int NominalVoltage,int NominalFrquency,
   int SwitchingFrequency, int AccelerationTime)
2.
3.   {
4.   Un = NominalVoltage;
5.   fn = NominalFrquency;
6.   fsw = SwitchingFrequency;
7.   AccTime = AccelerationTime;
8.
9.
10.  Un_pf_peak = 0.8165 * Un;//sqrt(2)/(sqrt(3))
11.  }
```

Code 6.1: Scalar Control Initialization code

The controllers Run code is presented in Code 6.2. Input values to the code are desired stator frequency and boost voltage. The V/f-gradient is calculated (line 4) and used to determine the amplitude of the voltage vector. From the code it can be seen that the frequency is ramped up and down according to the chosen ramp time. The boost voltage is introduced at line 33-42. The magnitude of the boost voltage is reduced as the speed is increasing. Lastly, the frequency is integrated using the actual stator frequency and switching frequency.

As mentioned earlier, the code will also go into field weakening operation when the desired stator frequency exceeds the nominal frequency. Field weakening code is not presented in this chapter. Up and down ramping of frequency is exactly the same as for normal operation, however voltage is kept constant.

```
1.  void ScalarControl::Run(int frequency, int BoostVoltage)
2.   {
3.           //Volt per hertz
4.           VoltPerHertz = Un_pf_peak/fn;
5.
6.           //Limits the frequency
7.           fref = frequency;
8.
9.           //Normal operation
10.          if (fs <= fn)
11.          {
12.          //Ramping up the voltage
13.          if (fref > fs)
14.                  {
15.                  fs += (fn/(fsw*AccTime));
16.                  U_s_amp = VoltPerHertz*fs;
17.                  }
18.
19.          //Ramping down the voltage
20.          else if (fref < fs)
21.                  {
22.                  fs -= (fn/(fsw*AccTime));
23.                  U_s_amp = VoltPerHertz*fs;
24.                  }
25.
26.          //Constant voltage
27.          else
28.                  {
29.                  U_s_amp = VoltPerHertz*fs;
30.                  }
31.
32.          //Boost Voltage at low speed
33.          if (fs == 0)
34.                  {
35.                  temp = 0;
36.                  }
37.          else
38.                  {
39.                  temp = (fn - fs) / fn;
40.                  U_boost = temp * (fmax(fmin(BoostVoltage, 10), 0));
41.                  U_s_amp += (U_boost);
42.                  }
43.          }
44.  //Sets the angle between 0-2*pi
45.                  U_s_ang += (fs/fsw*2*M_PI);
46.
47.                  if ( U_s_ang > 2*M_PI)
48.                          {
49.                          U_s_ang -= 2*M_PI;
50.                          }
```

Code 6.2: Scalar Control Run code

6.3 PWM (Modulator)

The PWM code is presented in Code 6.3. Voltage vector angle and amplitude along with DC-link voltage and blanking time gets read into the function. Voltage vector is decomposed to its respective α^s and β^s components (line 11-12). Using inverse Clark transformation, eq.(2.11), α^s, β^s components are converted to three phase voltages. The phase voltages are scaled according to the DC-link voltage ranging from -1 to 1 (line 19-21), where $-1 = -U_{dc}/2$ and $1 = U_{dc}/2$. Since the FPGA uses fixed point integer variables as input, control voltages are scaled with a scaling factor and type-casted to a 32-bit integer variable before being passed to the PWM IP-core, seen from line 21-27.

A load current is modulated in the code and passed down to the blanking time correction IP-core. Load currents are in phase with phase voltages. It will be shown in the next subchapters that blanking time correction is dependent on current direction and not amplitude, hence current amplitude is the same as control voltage amplitude. The integer typecasting and scaling is shown in line 29-31 and the value is written to the IP-core in line 32-34. Finally, blanking time interval is multiplied with sampling time period of the FPGA before being passed down to the correction IP-core.

```
1. void PWM_2L_3Ph::Run_3phase(float VoltageAmp, float VoltageAng,
   float DCLinkVoltage, float BlankingTimeIn)
2.   {
3.           //Declaring private variables
4.           U_s_amp = VoltageVectorAmp;
5.           U_s_ang = VoltageVectorAng;
6.           Udc = DCLinkVoltage;
7.
8.           //Alpha/Beta calculation
9.           U_s_alpha = U_s_amp * cos (U_s_ang);
10.          U_s_beta = U_s_amp * sin (U_s_ang);
11.
12.          //3 phase calculation
13.          U_s_a = U_s_alpha;
14.          U_s_b = 0.5 * (-U_s_alpha + sqrt(3) * U_s_beta );
15.          U_s_c = 0.5 * (-U_s_alpha - sqrt(3) * U_s_beta );
16.
17.          m_a = 2 * U_s_a / Udc;
18.          m_b = 2 * U_s_b / Udc;
19.          m_c = 2 * U_s_c / Udc;
20.
21.  // Convert to integer values:
22.  pwm_a=static_cast<int32_t>(fmax(fmin(m_a, 1.0), -1.0) * 2000);
23.  pwm_b=static_cast<int32_t>(fmax(fmin(m_b, 1.0), -1.0) * 2000);
24.  pwm_c=static_cast<int32_t>(fmax(fmin(m_c, 1.0), -1.0) * 2000);
25.          PWM_REFERENCE_REG(pwm_base_adresse,0) = pwm_a;
26.          PWM_REFERENCE_REG(pwm_base_adresse,1) = pwm_b;
27.          PWM_REFERENCE_REG(pwm_base_adresse,2) = pwm_c;
28.
29.          ia_out = m_a * 16383;  //Scale to a 16-bit value
30.          ib_out = m_b * 16383;  //Scale to a 16-bit value
31.          ic_out = m_c * 16383;  //Scale to a 16-bit value
32.          blankingtimecorr_ia_in_write(&ptr_blank, ia_out);
33.          blankingtimecorr_ib_in_write(&ptr_blank, ib_out);
34.          blankingtimecorr_ic_in_write(&ptr_blank, ic_out);
35.
36.  //Write BlankingTime to FPGA
37.  BlankingTime = BlankingTimeIn * 100E6; //Nr. FPGA samp per intr
38.  blankingtimecorr_blankingtime_write(&ptr_blank, BlankingTime);
```

Code 6.3:PWM Run code

6.4 Voltage Estimation with Blanking Time Correction

This chapter will present a method to estimate the output voltage of the inverter, U_{as}^*, using the DC-link voltage, knowledge of blanking time and current direction. An expression for U_{as}^* was thoroughly deducted and presented in eq.(4.34). The equation is repeated below.

$$U_{as}^* = \frac{1}{3} \cdot U_{dc} \cdot \left(\frac{2T_a^* - T_b^* - T_c^*}{2T_{samp}} \right) \tag{6.1}$$

Where: $T_a^* = T_a - \text{sing}(i_{sa}) \cdot t'$ $T_b^* = T_b - \text{sing}(i_{sb}) \cdot t'$ $T_c^* = T_c - \text{sing}(i_{sc}) \cdot t'$

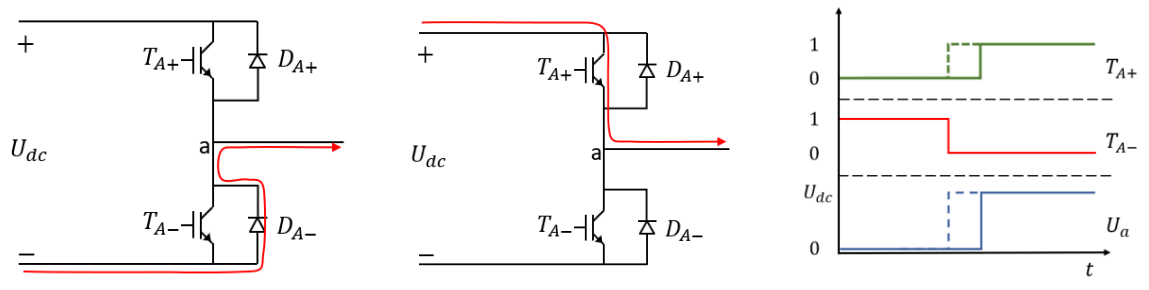
$t' = t_{off} - t_{on} - \Delta t$

In this model, on and off delays of the IGBT, t_{on} and t_{off}, are assumed to be equal and independent of current. However, switching waveform presented in section 4.1.1 shows that in real life applications, this will not be true. Current dependency could be included in the estimator when it is used on a physical drive. Moreover, t_{off} delay is much longer than t_{on} delay. This should also be taken into consideration when implemented into a real drive. Δt is the blanking time in a bridge leg. Blanking time is a constant time delay between turn-on and turn-off of the two complementary IGBTs and is used to prevent a short circuit of the bridge leg. In other words, when the upper IGBT turns off, turn-on signal for the lower IGBT is delayed according to Δt. Blanking time is a known time delay and is illustrated in Figure 4.4-(b).

From eq.(6.1) it is seen that actual on-time and sampling time has to be acquired in order to estimate phase voltage. Blanking time of IGBTs are given by the gate driver on the inverter's circuit board, thus making output signals from controller equal to the ideal switching presented in Figure 4.4-(a) and given by T_a, T_b, T_c. It is necessary to create additional logics in the FPGA in order to capture the ideal on-time and subtract or add the blanking time, in accordance with the equations above.

There are four scenarios one has to analyse to determine when and where subtraction or addition of blanking time has to occur in order to give a correct output voltage estimate. The scenarios are: 1) T_{A-} is opening and T_{A+} is closing with current flowing *to* the load, 2) T_{A+} is opening and T_{A-} is closing with current flowing *to* the load, 3) T_{A-} is opening and T_{A+} is closing with current flowing *from* the load, 4) T_{A+} is opening and T_{A-} is closing with current flowing *from* the load.

Figure 6.3 illustrates scenario 1). Figure 6.3-(a) shows the current flow when T_{A-} is closed. Since the IGBT is one-directional due to its internal diode, current has to go in the freewheeling diode D_{A-}, thus giving bridge leg midpoint the electrical potential of $0\ V$. During switching transaction, blanking time introduce a time interval where none of the IGBTs are conducting. The positive current continues to flow through D_{A-}, extending the time interval where electric potential of bridge leg a is $0\ V$. Since it is the on-time of T_{A+} that is being delayed, it can be understood that blanking time has to be subtracted in order to find T_a^*. This delay is illustrated in Figure 6.3-(c) where the dotted line shows the ideal on-switching and the solid line shows the actual on-switching.



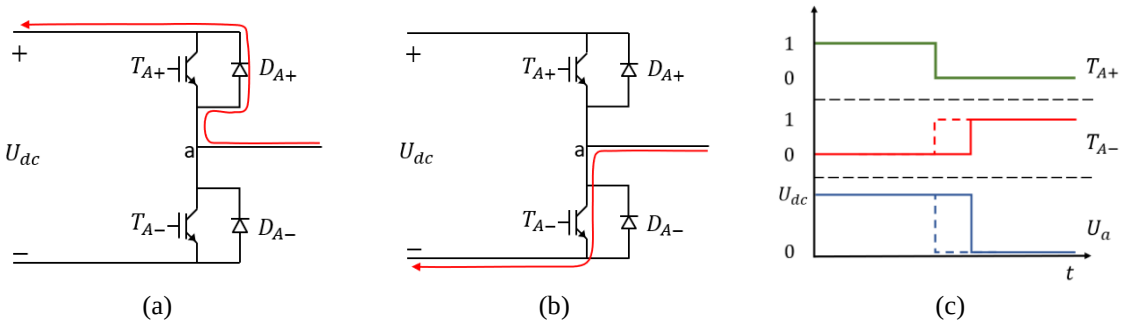
(a) (b) (c)

Figure 6.3: Switching behaviour with blanking time, positive current

Scenario 2) can also be illustrated using Figure 6.3. Initially T_{A+} is closed and current flows through the IGBT, as shown in Figure 6.3-(b). Electric potential at the midpoint will then be equal to U_{dc}. As for the first scenario, during switching transaction, blanking time introduce a time period where none of the IGBTs are closed. Positive current will immediately commutate from T_{A+} to D_{A-} when T_{A+} opens, changing the electrical potential of the bridge leg from U_{dc} to $0\ V$ (Figure 6.3-(a)). Since the electrical potential follows the ideal switching, no correction is done in scenario 2).

It can be seen from Figure 6.4 that scenario 3) becomes similar to scenario 2). Current commutate immediately from T_{A-} to D_{A+} when the lower IGBT is opens. Midpoint potential immediately rise from $0\ V$ to U_{dc} and blanking time correction is not preformed.

Scenario 4) can also be analysed using Figure 6.4. From Figure 6.4-(a) it is seen that current flows through D_{A+} when T_{A+} is conducting. During the blanking time interval with no conducing IGBTs, the negative current will still flow in the upper diode, extending the duration where electrical potential is equal to U_{dc}. This extension is illustrated in Figure 6.4-(c) where ideal voltage is given dotted line and actual voltage is given by the solid line. This increased voltage area shows that blanking time has to be added to T_a after T_{A+} has closed in order to find T_a^*.



(a) (b) (c)

Figure 6.4: Switching behaviour with blanking time, negative current

One way of estimating T_a^* between two CPU interrupts is to count the numbers of clock cycles in the FPGA where the bridge leg has an electrical potential equal U_{dc}. The analysis above showed that the time duration with this potential was shorter than expected when the current was positive, i.e. flowing to the load, and longer when the current was negative, i.e. flowing from the load. Moreover, the analysis also showed that loss of voltage area happened when T_{A+} was supposed to closed (rising edge control bit) and an addition of voltage area happened when T_{A+} was opened (falling edge control bit). By detecting rising and falling edges of the control bit alongside with direction of current, it is possible to write a FPGA program that counts the number of cycles where electrical potential of the bridge leg is equal to U_{dc}.

An example of such a program is presented in Figure 6.5. The program is made using Xilinx library in Simulink. Input signal are: bit signal to the upper IGBT in the bridge leg, phase current, blanking time and one reset signals and one enable signal given by the interrupt. Input signals are presented with a green background colour. Output signal is the number of clock cycles with midpoint potential equal U_{dc} and is coloured blue. The program is built using logical ports, counters, delays and registers. The upper violet area shows the logics for scenario 1) and 2) and the lower area shows the logics for scenario 3) and 4). The different segments of the program will be discussed in detail below.

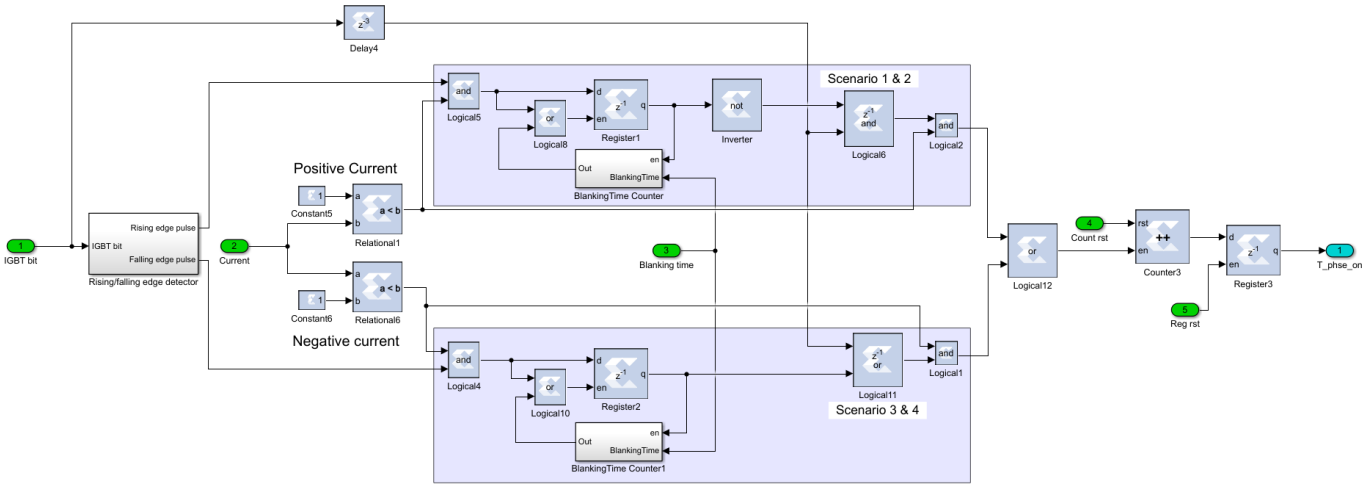


Figure 6.5: Blanking time compensator

The IGBT control bit is connected to a rising/falling edge detector that sends out a positive pulse to ether *rising edge pulse* or *falling edge pulse*. The content of the detector and a plot of its input and output signal can be seen in Figure 6.6.

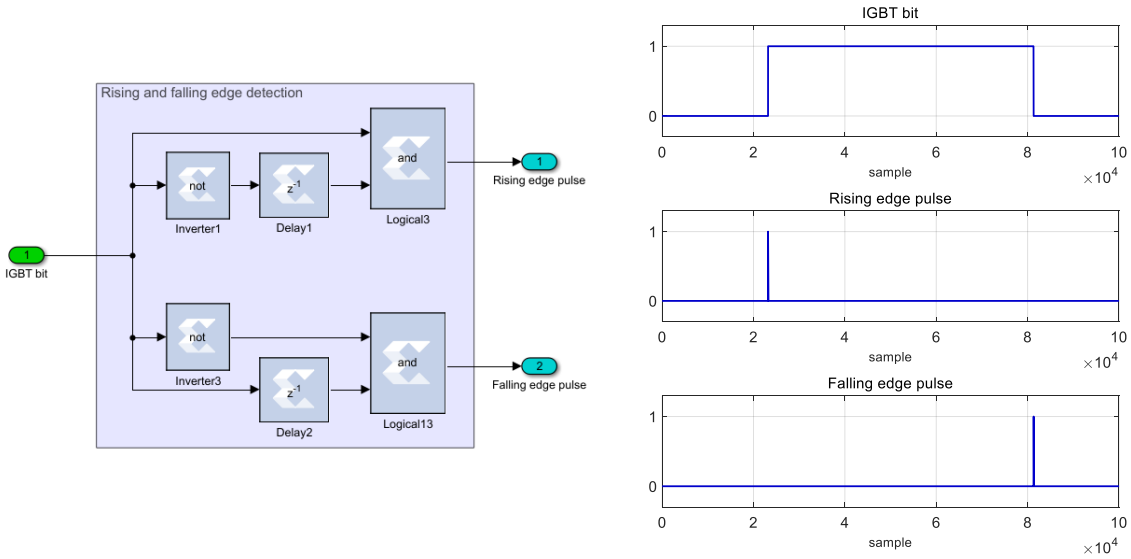


Figure 6.6: Rising and falling edge detector

Current direction is found by comparing the current signal with 0. Output of the comparator gives a constant value of 1 in either positive or negative current branch. A magnified picture of the comparator and plots of input and output signals are shown in Figure 6.7.

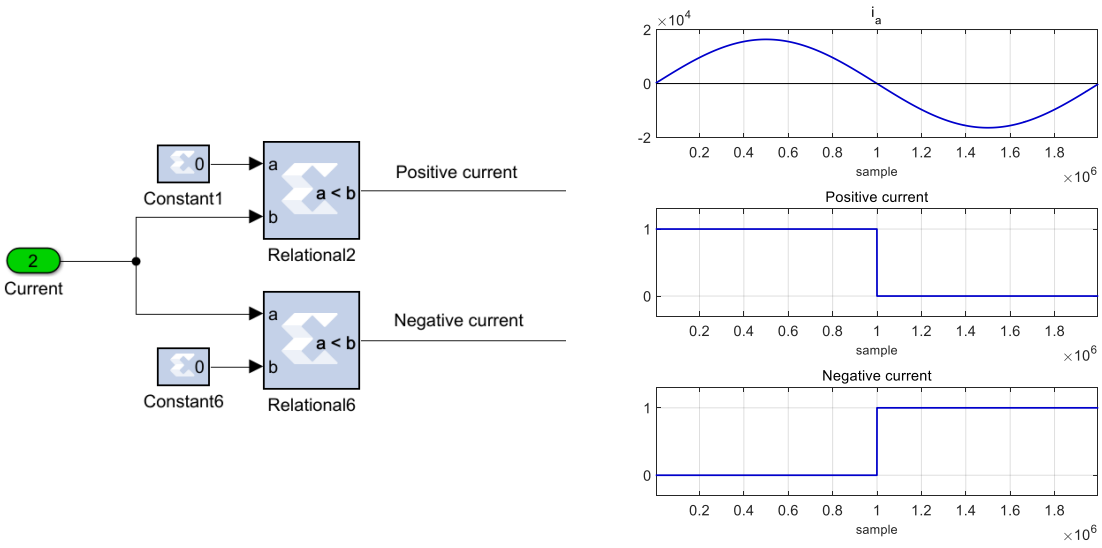


Figure 6.7: Sign of current

The counter logic used for scenario 1) is presented in Figure 6.8. The rising edge pulse and positive current signal sends a pulse through the AND port, shown in Figure 6.8-(c). The pulse both enables and latches into the register, making the register output, *q*, equal to 1. The register output will continue to have the value 1 until it is enabled again and reads in an input value, *d*, equal to zero. Output of the register is connected to an enable port of a counter that starts counting. The sampling period of the FPGA is 10 *ns*. The counter is increasing the value by one for each sample. In order to get the desired blanking time, it is therefore necessary to divide the blanking time with sampling period. This deviation is done in the CPU and is shown in Code 6.3, line 37. When the counter exceeds this value, it resets itself, shown in Figure 6.8-(e), and enables the register. It can be seen from Figure 6.8-(c) that *d* is zero, thus making *q* equal to zero. Figure 6.8-(d) show that *q* is high during the time instance between the rising edge pulse and the blanking time counter reaching its value.

Similar behaviour is found when the current is negative and a falling edge pulse is detected, as for scenario 4).

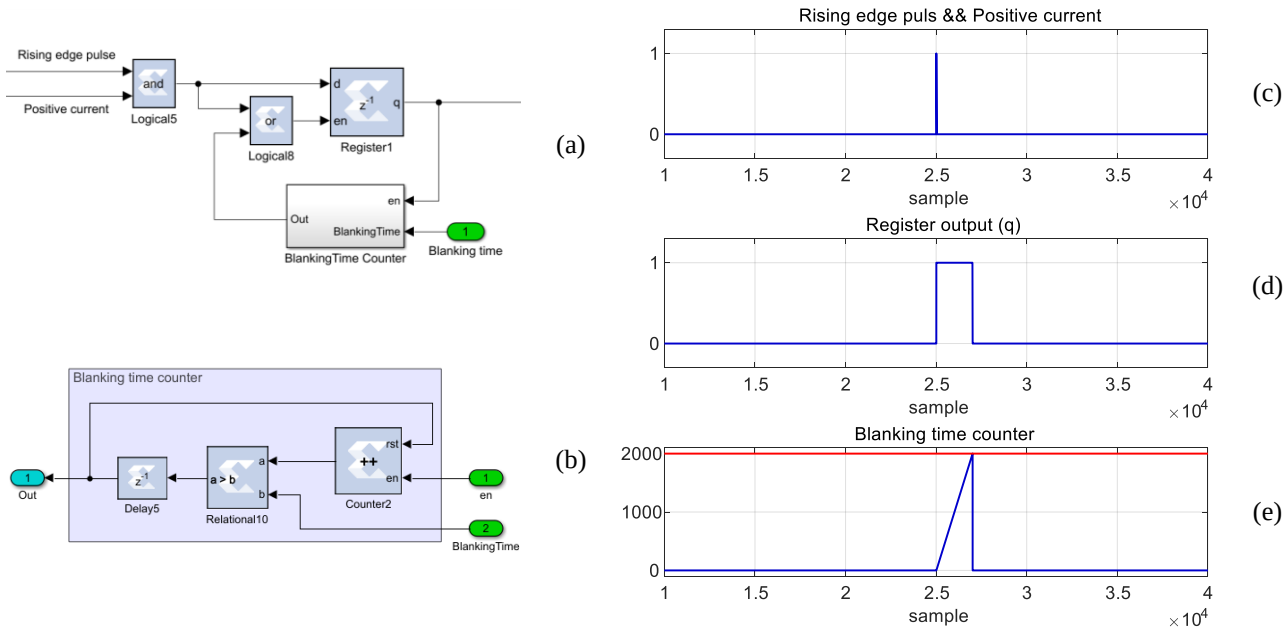


Figure 6.8:Blanking time counter

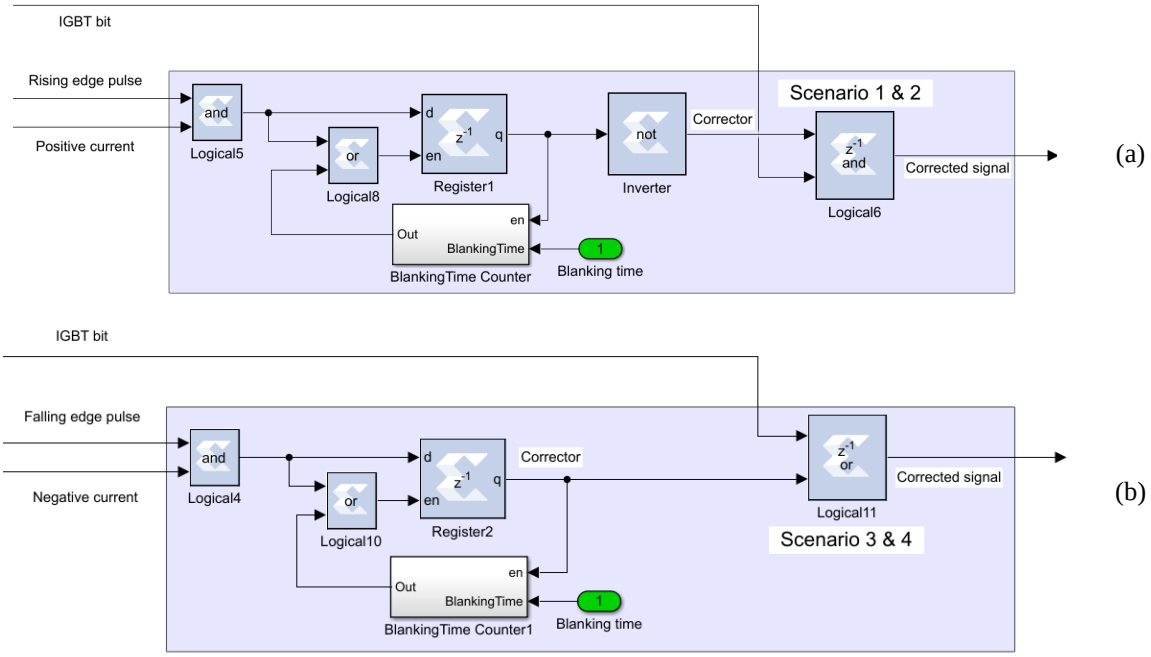


Figure 6.9: Blanking time correction of IGBT bit. (a) positive current. (b) negative current.

The correction logic used for positive and negative current is presented in Figure 6.9-(a) and -(b), respectively. Time-plots of Figure 6.9-(a) are presented Figure 6.10-(a). The positive current and rising edge pulse sets the register output high. This signal is inverted and used in a AND operator together with the IGBT bit. It is seen from Figure 6.10-(a) that the corrected signal is delayed with the inversed register output when the IGBT has a rising edge and follows IGBT bit during falling edge, in accordance with analysis done for scenario 1) and 2).

Time-plots for Figure 6.9-(b), i.e. negative current, are presented Figure 6.10-(b). From the analysis of scenario 3) and 4) it was shown that the corrected signal should follow the IGBT bit during a rising edge signal and extend the time period to which the signal was high after a falling edge signal. Negative current and falling edge pulse sets the register output high. The output stays high until banking time counter has reached its target value. It can be seen from Figure 6.10-(b) that by using a OR operator, register output works as an extension of the IGBT bit.

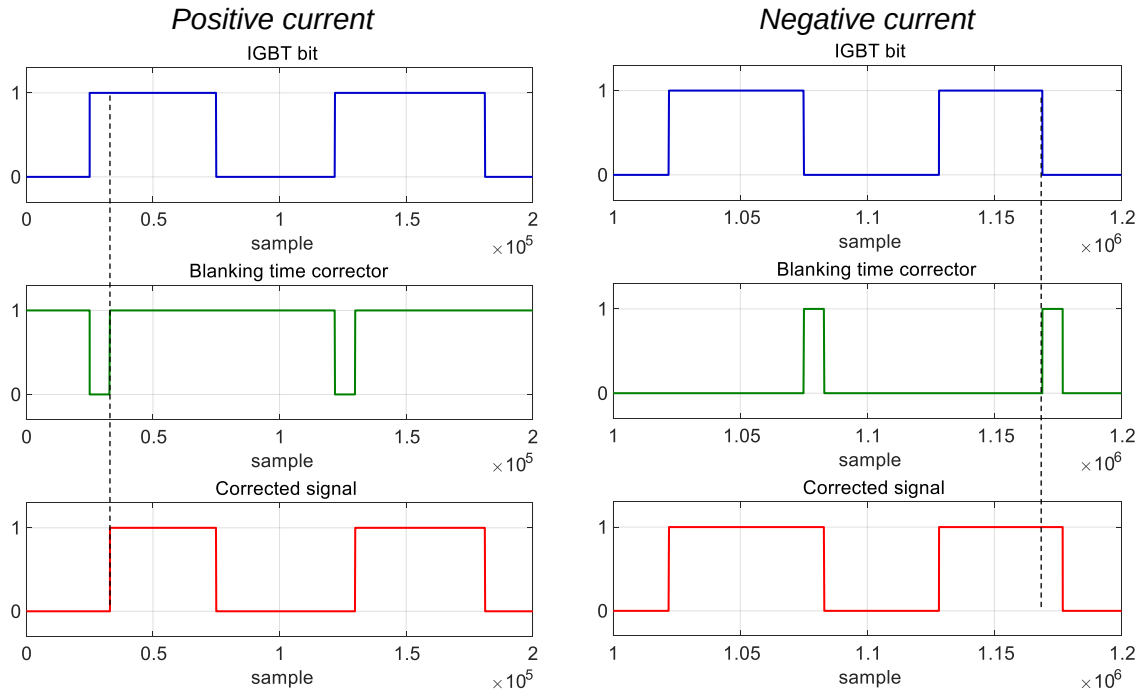


Figure 6.10: Time-plot of corrected and uncorrected IGBT signal

Output of scenario 1) and 2) are connected to an AND gate together with positive current signal from Figure 6.7. This ensures that the delayed "on time" only occurs when the current is positive. The same goes for scenario 3) and 4). The two correction pathways are then combined with help of an OR gate and connected to the enable port of a counter which again is connected to a register. This logical circuit is presented in Figure 6.11-(a). The counter is latched into the register and reset at every interrupt. The corrected bit and interrupt signal, on-time counter and register output are presented in Figure 6.11-(b). Blanking time makes the signal unsymmetrical round the interrupt. This asymmetry is reflected in the counter and register. Finally, in can be understood that the integration of the corrected bit, i.e. output of the counter, becomes an estimate of the corrected on -time, T_a^*. This value is latched into the register.

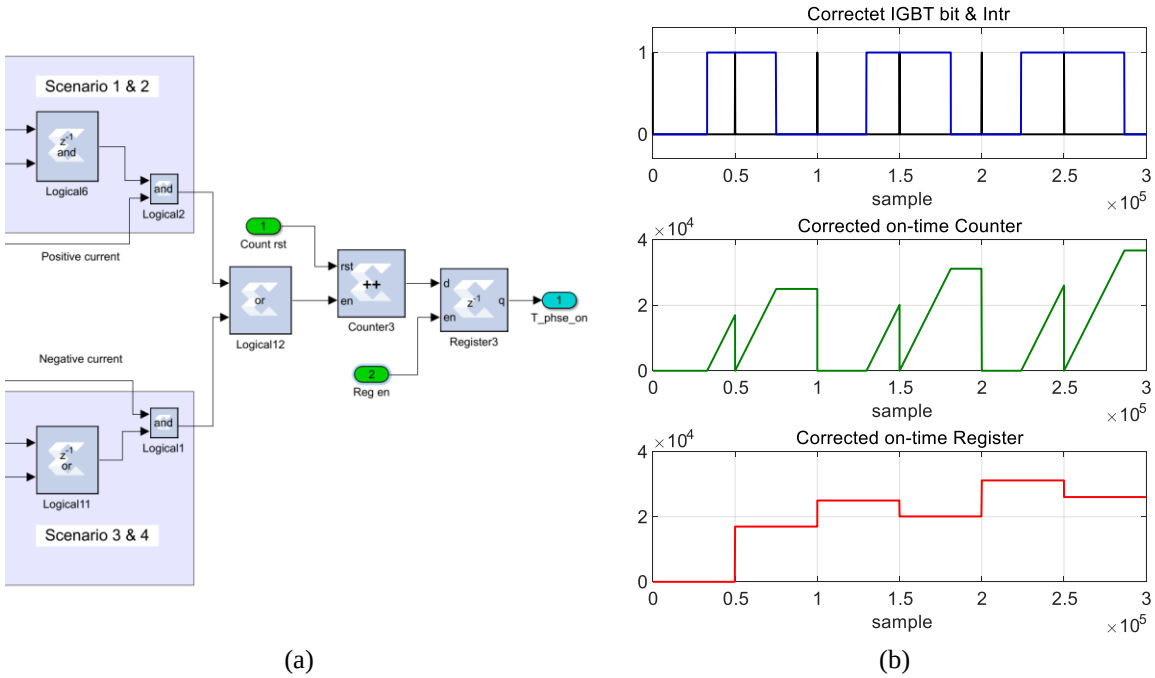


Figure 6.11: (a) counter and register logic. (b) corrected signal, on-time counter and on-time register

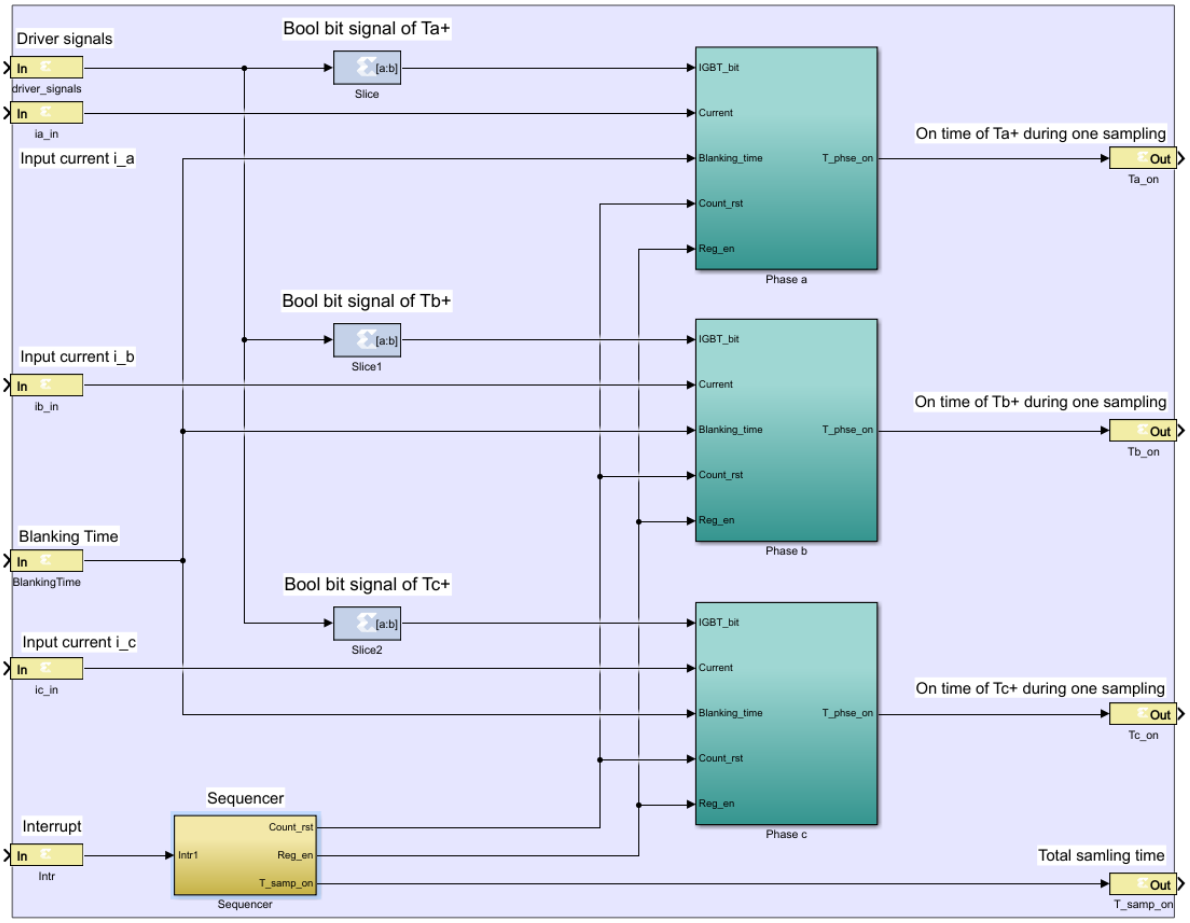


Figure 6.12: Complete block design of the blanking time corrector IP-core

Until now, the subchapter has presented a logical circuit that can be used in order to find the corrected on-time of one bridge leg of the inverter. The two other bridge legs can be found using the same circuit. The total block design of the blanking time corrector is presented in Figure 6.12, where the input gates of the block is located to the left and the output gates are located at the right. The circuit shown in Figure 6.5 is made into a subsystem and is used to calculate the actual on-time of all the three bridge legs. Driver signal input (IGBT bit input) is connected internally in the FPGA to a driver interface IP-core which again is driven by the PWM-IP. The driver interface sends out six bits, one for each IGBT. The bit for the upper IGBT of the respective bridge leg is picked out of the bit-word using a slice operator. Input ports ia_in, ib_in, ic_in, BlankingTime and output ports Ta_on, Tb_on, Tc_on, T_samp_on are all defined as AXI4-Lite ports, thus allowing communication with the CPU.

The sequencer, shown in the bottom left corner of Figure 6.12 and in details in Figure 6.13, is driven by the interrupt signal generated by the PWM IP-core. The signal is type-casted to a Boolean signal and connected to a rising edge detector. When a rising edge of the interrupt is detected an enable signal is sent to the on-time registers, shown in Figure 6.11, latching in the latest value of T_a^*, T_b^*, T_c^* to the register output which in terms are read by the CPU. The on-time counters are also reset. It can be seen that the reset signal is delayed with 11 clock cycles. This is done to ensure the counter is not reset before the value is latched into the register which can happen due to internal rooting delays in the FPGA. The interrupt also drives a counter that counts the total number or FPGA samples between two interrupts. The counter and register circuit are similar to the one presented in Figure 6.11. However, the fixed interrupt frequency makes the output value of the register constant. This constant value is read by the CPU and used as T_{samp}.

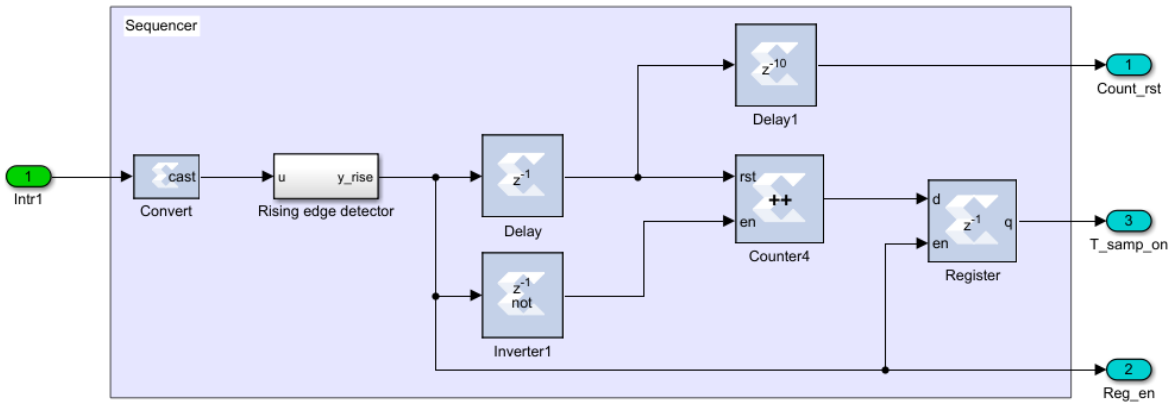


Figure 6.13: Sequencer block

Knowing T_a^*, T_b^*, T_c^*, it is possible to calculate the actual phase voltage of the load connected to inverter terminals using eq.(6.1). Code for the voltage estimator is presented in Code 6.4. On-time register output of the three phases and total count between two interrupts are read from the IP-core (line 4-7) using the "read function" described in chapter 5.3. The calculation of phase voltages is shown in line 11-21. It can be seen that eq.(6.1) is somewhat modified in order to give the average voltage over a sampling period instead of one switching period.

```
1. void VoltageEstimator::Run(int DCLinkVoltage)
2.   {
3.   //Get nr. of samples from FPGA
4.   N_T_intr = (float) blankingtimecorr_t_samp_on_read(&ptr_blank);
5.   N_Ta_on = (float) blankingtimecorr_ta_on_read(&ptr_blank);
6.   N_Tb_on = (float) blankingtimecorr_tb_on_read(&ptr_blank);
7.   N_Tc_on = (float) blankingtimecorr_tc_on_read(&ptr_blank);
8.
9.
10.  //Voltage calculation according to formula
11.  N_on_a_phase = 2 * N_Ta_on - N_Tb_on - N_Tc_on;
12.  N_on_b_phase = 2 * N_Tb_on - N_Ta_on - N_Tc_on;
13.  N_on_c_phase = 2 * N_Tc_on - N_Ta_on - N_Tb_on;
14.
15.  T_on_a_phase = N_on_a_phase/N_T_intr;
16.  T_on_b_phase = N_on_b_phase/N_T_intr;
17.  T_on_c_phase = N_on_c_phase/N_T_intr;
18.
19.  U_sa = DCLinkVoltage * 0.3333 * T_on_a_phase;
20.  U_sb = DCLinkVoltage * 0.3333 * T_on_b_phase;
21.  U_sc = DCLinkVoltage * 0.3333 * T_on_c_phase;
22.  }
```

Code 6.4: VoltageEstimator code

6.5 Distortion voltage correction

It was shown in chapter 4.1 that blanking time introduced a six-step trapezoidal voltage that distorted the output voltage of the inverter. It was discussed in chapter 4.2 how the distorted voltage affected both open loop V/f-control and closed loop FOC. It was understood that the trapezoidal distortion voltage created a discontinuous output voltage when the machine was controlled using V/f-control and a similar distortion in control voltage when the machine was controlled using FOC.

It has been shown that the distorted voltage introduced to the output voltage could be expressed by eq. (4.35). The distorted voltage equation is repeated below:

$$U'_{sa} \approx \frac{1}{6}\left(\frac{U_{dc} \cdot t'}{T_{samp}} - U_{sw0} + U_{fd0}\right)\left(2\text{sign}(i_{sa}) - \text{sign}(i_{sb}) - \text{sign}(i_{sc})\right) \tag{6.2}$$

By introducing the same distorted waveform to the control voltage, but with opposite sign as observed in the output voltage, it is possible to compensate for this distortion. Such correction for the non-linear voltage drop has been proposed in several papers [10, 19, 24]. The authors of [24] propose to use a PI-regulator to tune the amplitude of the correction voltage. Tuning of amplitude is necessary if turn-on and -off delays and forward voltages are unknown. In this thesis, the control board does not control any physical inverter, thus turn-on and -off time together with forward voltage of the switch and power diode is set equal to zero. Since all variables of eq.(6.2) are known, compensation voltage should not need tuning in order to remove voltage distortion from the output voltage.

The correction code is presented in Code 6.5. The correction code is implemented in the already existing modulator code presented in Code 6.3. Blanking time and T_{samp} is read from the blanking time IP-core (line 8-9). The correction voltage for the different phases is calculated according to eq.(6.2) (line 11-13). Finally, correction voltage is added to the control voltage (line 16-18).

```
1. void PWM_2L_3Ph::Run_3phase(float VoltageAmp, float VoltageAng,
   float DCLinkVoltage, float BlankingTimeIn)
2.   {
3. …
4. …
5.
6. if (VoltageCorrectionOn == 1)
7. {
8. //Reading blanking time and T_samp from FPGA
9. blank_time = 1E-8*blankingtimecorr_blankingtime_read(&ptr_blank);
10.  T_samp = 1E-8*blankingtimecorr_t_samp_on_read(&ptr_blank);
11.
12.  U_a_InvCorr = 0.16667 * ((Udc*(blank_time))/T_samp) *
                    (2*sign_a - sign_b - sign_c);
13.  U_b_InvCorr = 0.16667 * ((Udc*(blank_time))/T_samp) *
                    (2*sign_b - sign_a - sign_c);
14.  U_c_InvCorr = 0.16667 * ((Udc*(blank_time))/T_samp) *
                    (2*sign_c - sign_a - sign_b);
15.
16.  //Modulation indexes with non-linear correction
17.  m_a = (m_a + (2*U_a_InvCorr/Udc));
18.  m_b = (m_b + (2*U_b_InvCorr/Udc));
19.  m_c = (m_c + (2*U_c_InvCorr/Udc));
20.  }
21.  …
22.  …
```

Code 6.5: DistortedVoltageCorretion code

7 Program Verification through Simulation and Testing

7.1 Introduction

The program has been tested and debugged using a combination of Simulink simulations and the real-time monitoring tool “WatchView”, provided by The Switch. Simulink has mainly been used in the work regarding the blanking time corrector IP-core. Simulink simulation has been used extensively during development, testing and debugging of the new IP-core and has proven to be a very powerful and efficient tool. WatchView has been used to monitor and run the control board. WatchView communicates with the CPU and allows parameters within the CPU program to be changed and monitored. The software has been used during development and testing of all C++ routines that have been written or modified. It is not possible to use WatchView to monitor and control signals in the FPGA. However, input and output values from the IP-core can be accessed if the gates are defined as AXI4-Lite and properly initialized in the CPU software. Since WatchView communicate with the CPU, values from the FPGA can only be written to or read from during each CPU interrupt. This makes WatchView suitable for testing and verifying IP-cores and not suitable for debugging.

The goal of this chapter is to verify the changes and additions done in the programming software. The chapter will focus on the blanking time correction IP-core and the C++ programs related to it. Print screens of time-plots from WatchView will be compared to simulation results from Simulink.

In order to conduct proper simulations, a PWM model has been provided. The PWM model is made by Anirudh Budnar Acharya (PhD. candidate at NTNU) and is shown in Figure 7.1. Control voltage is changed using the sinewave generator to the left in the figure. Control voltage input are updated each interrupt and the value is compared with a triangular wave. The sampling frequency of the model can be changed between equal to the triangular wave or half the triangular wave. Upper IGBT-bit for the three phases are given by Sa_out, Sb_out and Sc_out. These outputs are connected directly to the IGBT-bit in-port seen in Figure 6.12. The PWM modulator also provide an interrupt signal according to the sampling time which is connected the sequencer in Figure 6.12.

The current signal is generated using the same circuit as the control voltage and the blanking time can be controlled using a Constant block.

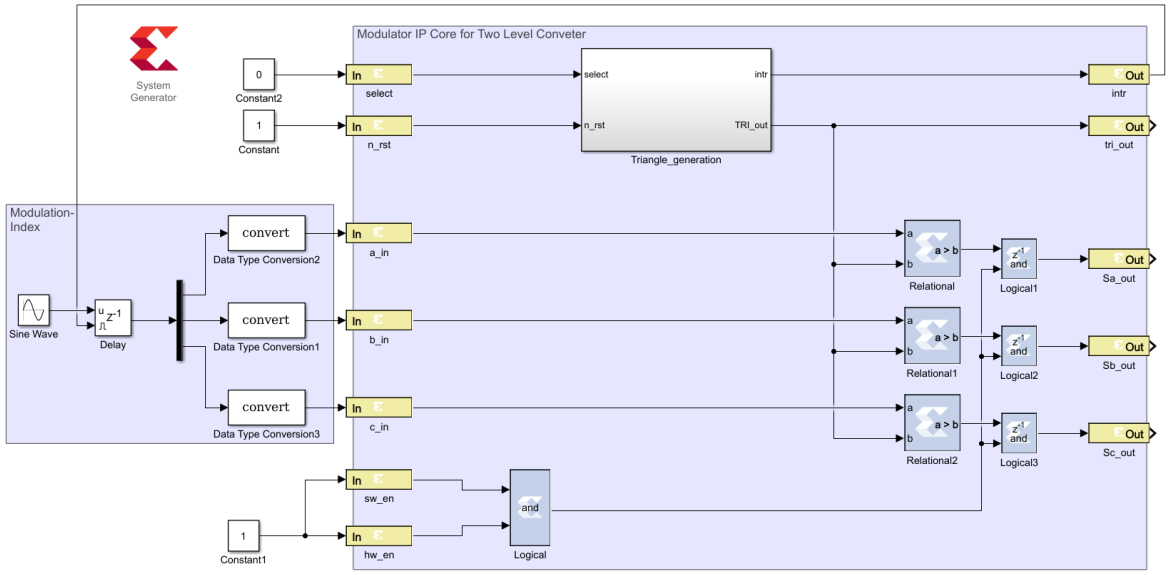


Figure 7.1: PWM model

7.2 DC-voltage test

To verify that the blanking time IP-core is able to introduce a turn-on and turn-off delay to the IGBT control signal from the modulator as a function of phase current, a DC-voltage test is conducted to one of the bridge legs. DC-voltage is achieved by applying a constant control voltage. The bridge leg then acts as a half bridge converter, with operation in 1st and 4th quadrant. The duty cycle of the converter is set to 0.5, i.e. control voltage equal to 0. The DC-voltage test makes it easy to spot the increased and decreased on-time of the corrected IGBT bit. The constant control voltage should give a constant average output voltage of half the DC-link voltage, thus constant on-time of the IGBT bit. However, if the blanking time corrector works as intended, average voltage and on-time is expected to be reduced during positive current and increased during negative current.

The triangular wave and control voltage are presented in the first subplot of Figure 7.2. When the control voltage value is greater than the triangular wave, IGBT bit is set high and vice versa. An interrupt is given at every top and bottom of the triangular wave. The corrected and uncorrected IGBT bit together with the interrupt pulse are shown in the second subplot of Figure 7.2. In a real-life application, blanking time is expected to last somewhere between 1 and 3 μs. However, in order to illustrate the correction, blanking time has been specified to 20 μs during the DC-voltage test.

The triangular wave has a frequency of 1 kHz and the FPGA has a sampling frequency of 100 MHz. Since the interrupt happens every half period, the FPGA will sample $50x10^3$ times within two interrupt signals. The chosen control voltage is expected to create an IGBT bit signal that is high during half the sampling interval, i.e. $25x10^3$ samples. The IGBT bit given by the modulator is shown by the dashed green line in Figure 7.2. The figure shows a symmetrical on-signal around the interrupt pulse. Looking at the corrected IGBT bit, shown by the blue line in Figure 7.2, it is observed that blanking time introduce a turn-on and turn-off delay depending on the current direction. The phase current is shown in the third subplots of Figure 7.2. As described in chapter 6.4, positive current (i.e. current flowing to the load) introduce a turn-on delay and negative current introduce a turn-off delay. The duration of the delays is given by T_{err} (from chapter 4.1.2) which in this thesis is equal to blanking time. By dividing the blanking time on the FPGSs sampling time period, 10 ns, it is found that a blanking time of 20 μs corresponds to 2000 samples.

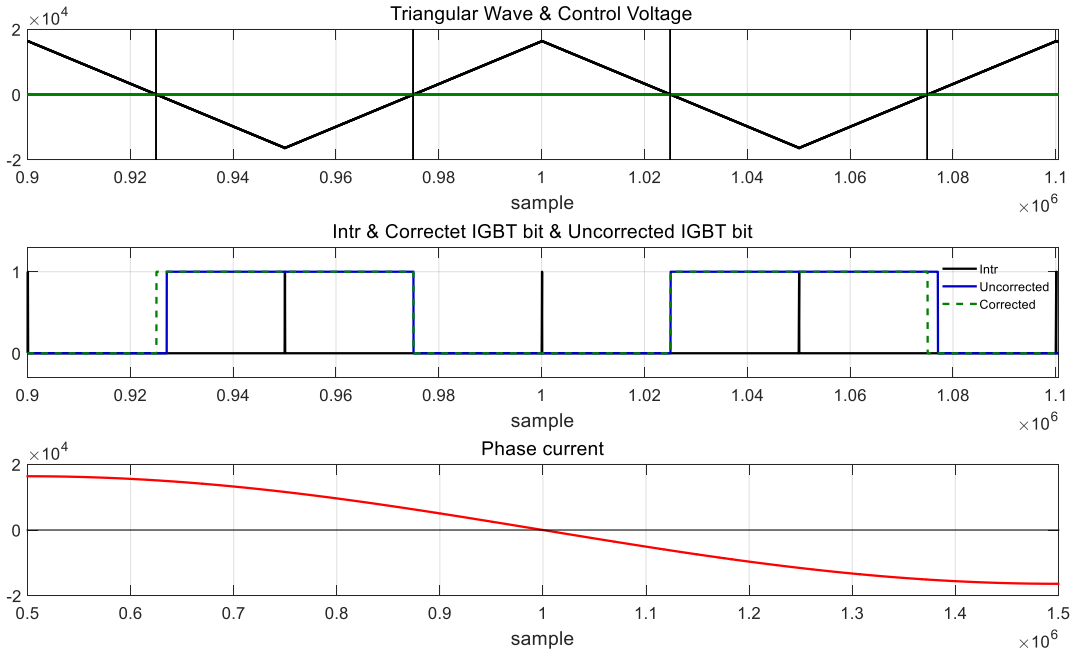


Figure 7.2: Illustration of Corrected and Uncorrected IGBT bit

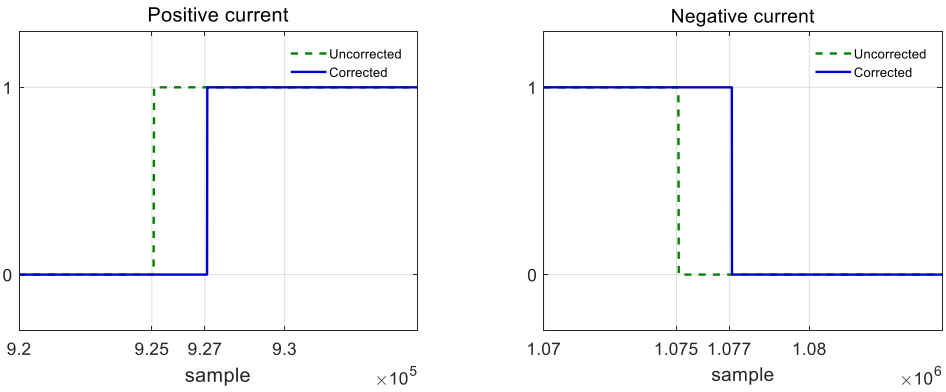


Figure 7.3: Close-up picture of turn-on and turn-off delay imposed by blanking time

A close-up picture of the turn-on and turn-off delays are shown in Figure 7.3. The figure shows the expected delay of 2000 samples with both positive and negative current. Going back to Figure 7.2, it can be understood that numbers of samples where corrected IGBT bit is high is expected to be $23x10^3$ between the two first interrupts and $25x10^3$ between the 2nd and 3rd. Similarly, when the current changes direction, numbers of samples from the 3rd to the 4th interrupts should also be $25x10^3$ and the number of samples between 4th and 5th interrupt should be increased to $27x10^3$.

Figure 7.4 shows the output value of the on-time count registers when the IP-core is simulated in Simulink. The first subplot shows the on-time count of the uncorrected IGBT bit from the modulator. As expected, the plot shows a constant value of $25x10^3$ on-time samples between two interrupts. The second subplot shows the on-time register of the corrected IGBT bit. The simulation result shows that the IP-core is able to introduce correct turn-on and -off delays. The results show a reduction in counted value of every other count according to the specified blanking time when the current is positive and similarly an increase in count when the current changes direction.

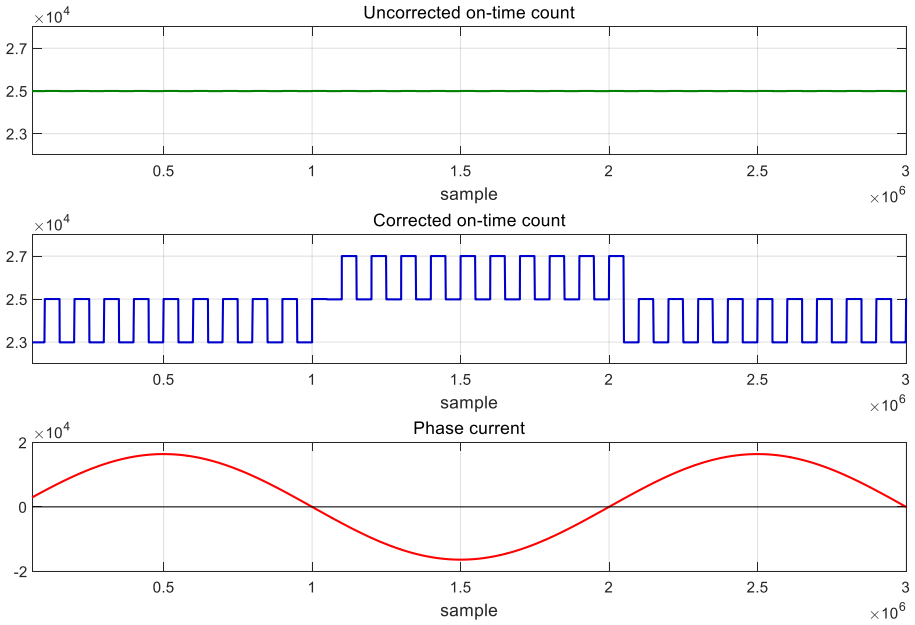


Figure 7.4: Simulation - on-time count for DC-test with and without blanking time

The same DC-voltage test is also performed on the control board. In the simulated case, counted on-time within two interrupts were presented for both the corrected and uncorrected IGBT bit. The control board is tested with and without a specified blanking time. This should give similar results as the simulations. Blanking time is specified to be 20 μs or 2000 samples, as for the simulation. Although tests of the control board and simulations seems to be quite similar, it is important to stress that where the simulation only tests the developed IP-core, control board tests requires the complete software platform to be operational in order to work. This includes a complete IP-core design for the FPGA, an operating C++ program for the CPU and communication between them. Another important difference is the sampling delays of signals. This will be discussed further later in this chapter.

Figure 7.5 show a print screen of the test results when the system is not subjected to blanking time. Dots on the lines represent the values used in the CPU for that interrupt. The program itself automatically draws a straight line between the values used by the CPU in order to illustrate waveforms easier.

As discussed earlier and shown in Figure 7.2, it is expected that the test shows a constant counted value. However, looking at the test results it is obvious that this is not the case. The value cursors show a difference of 126 samples between two samples. Furthermore, the total count of samples between two interrupts showed a constant value of 50554 samples. This indicate that the triangular wave used in the PWM IP-core has a frequency that is a bit lower than 1 kHz. As for the simulated case, it is expected that a constant control voltage of 0 without correction of blanking time will give an on-count of half the total counted value, i.e. 25277 sample. From Figure 7.5 it can be seen that the expected result of 25277 samples are in the middle of the counted values read from the IP-core. This indicate that the IGBT pulse is not symmetrical around the interrupt or that somewhere in the complete IP-core design it exists a delay on the IGBT pulse that the author is not aware of. It is seen from the print screen that the delay is independent of current direction.

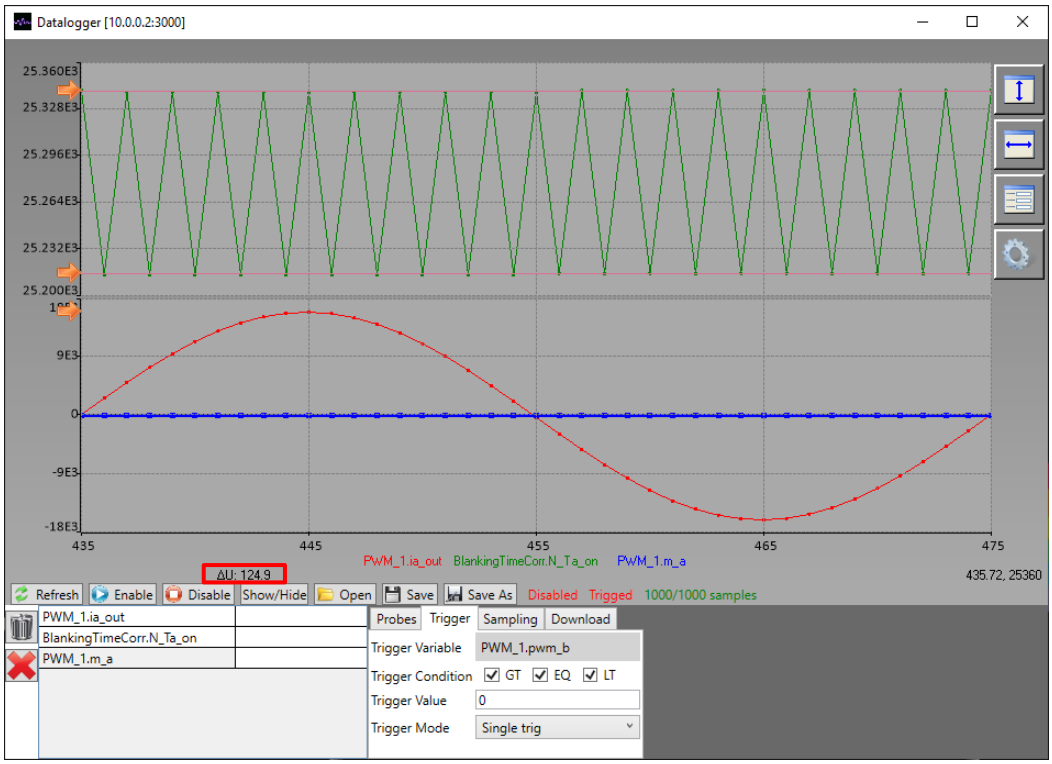


Figure 7.5:Control board - on-time count for DC-test without blanking time

Figure 7.6 shows a print screen of the test results when the system is subjected to a blanking time of 20 μs or 2000 samples. Value cursors shows an increase in the counted value of every other count by 2000 when the current is negative and a decrease in counted value when the current is positive. The test shows that the blanking time corrector IP-core is able to delay the on- and off-times according to the specifications discussed in chapter 6.4. The test also shows that the read and write functions provided by the System Generator tool works as anticipated.

The difference in on-time counted values that was found when the system was tested without blanking time can also be seen when blanking time is introduced. The deviation is best seen between sample number 474 and 475. Tests using the value cursor show that the two points are 126 samples apart. The added deviation is also seen in the on-time count of a particular current direction. Top and bottom values of the counted value were found to be 2126 samples.

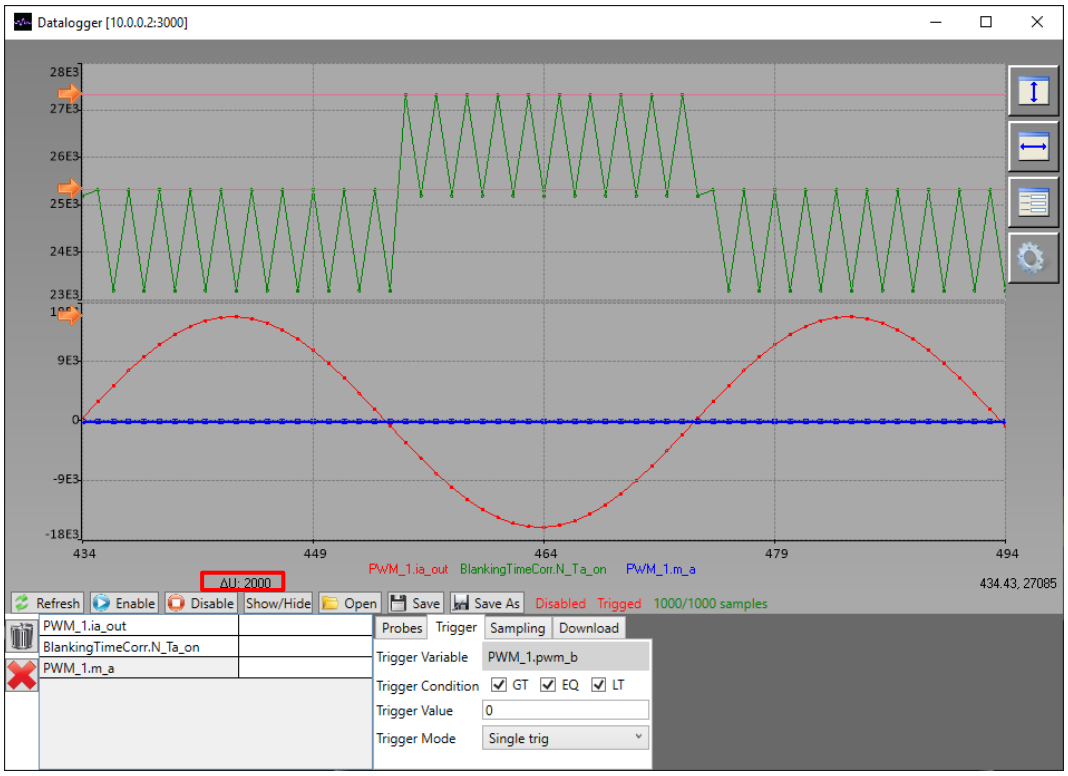


Figure 7.6 Control board - on-time count for DC-test with blanking time

7.3 AC-voltage test

Previous subchapter showed that the blanking time IP-core was able to introduce a turn-on and turn-off delay to the IGBT bit sent from the modulator, depending on the current. It also showed that the delay corresponded to the specified blanking time. The next step in verifying the program software is to conduct a voltage estimation test. As for the previous subchapter, this chapter will present simulations of the isolated IP-core and tests using the complete setup of the control board. Both simulations and control board test will be carried out with and without blanking time.

As discussed in chapter 4.2, distorted voltage and its effect on the output voltage of the inverter is independent of frequency. From chapter 3.1 it is known that during V/f-control, voltage is reduced linear with frequency. From this it can be understood that the distorted voltage has greatest effect on the output voltage when frequency is low, i.e. motor is running on low speed. It is in low speed operation area voltage estimation becomes most critical. During low speed operation it is expected that blanking time will introduce a sixth-harmonic distortion to the output voltage if the control voltage is kept sinusoidal, similar to what is shown in Figure 4.5. The simulation and tests are done with an output frequency of 2 Hz. The low frequency should cause a visual distortion on output voltage when blanking time is introduced. Blanking time is specified to be 2 μs or 200 samples for both control board tests and simulations.

Simulation results of the IP-core is presented in Figure 7.7. The figure shows control voltage and estimated output voltage, with and without blanking time. When blanking time is set to zero, the result shows an estimated phase voltage equal to the control voltage. As for the DC-voltage test, it is seen that that the IP-core does not correct the IGBT bit sent from modulator when the system is without blanking time. Phase voltage is calculated similar to what is shown in Code 6.4. The fact that control voltage and estimated voltage are equal shows that the IP-core is able to count correct on-time of all three phases and also the total time between two interrupts.

When the blanking time is specified to 2 μs it can be seen that the estimated voltage reduces in amplitude and becomes discontinuous. The DC-voltage test showed that for an isolated bridge leg, blanking time reduced the average output voltage when the current was positive and increased it when the current was negative. However, as shown in chapter 4.1, phase voltage of a 3-phase load connected to an inverter is not given exclusively by the bridge-leg voltage to which it is connected. The influence of the other bridge legs results in a non-linear voltage drop in the phase voltage. Blanking time will always reduce the voltage available on the output terminals of the load. The waveform of the

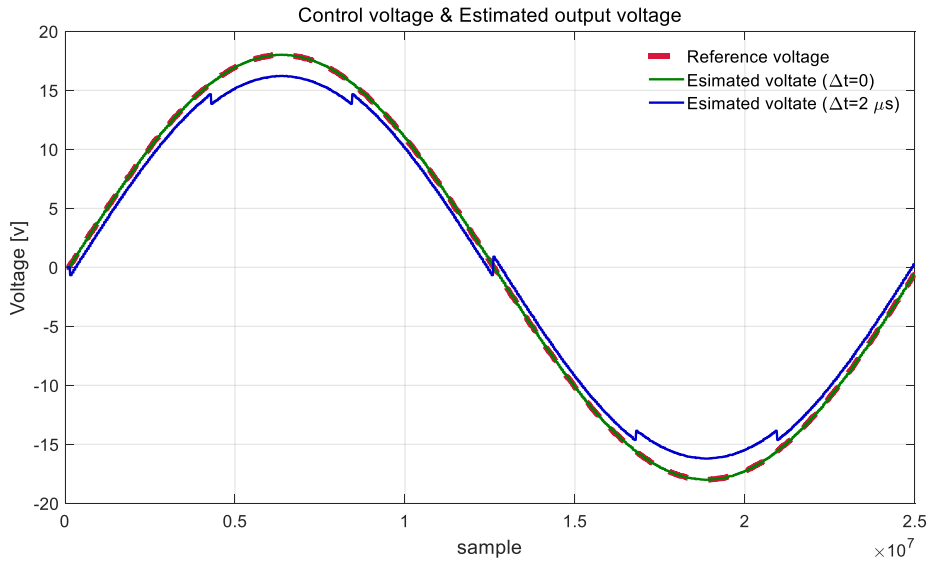


Figure 7.7: Simulation – Voltage estimation with and without blanking time

non-linear drop is shown in chapter 4.2 and corresponds to the simulation result. Figure 7.8 shows a print screen of the test result from the control board. The red signal shows the reference voltage controlled by the V/f-controller and the green signal shows the estimated output phase voltage. The test is conducted with blanking time specified equal zero.

As for the simulated test, the result shows an estimated phase voltage equal to the control voltage. Again, this shows that blanking time IP-core does not introduce any delay on the IGBT bit before counting the on-time when blanking time is equal to zero. Moreover, the fact that control voltage and estimated voltage are equal, verifies the transformations and scaling done in the controller and modulator (Code 6.1, Code 6.2 and Code 6.3), and communication between FPGA and CPU.

During testing, it was found that the system introduces a 2-sample delay between control voltage and estimated voltage. Control voltage used in the PWM IP-core uses one sample to read the value given by the CPU and the on-time counter counts for the following sample before sending the counted vale back to the CPU. Since the sampling frequency of the CPU is constant, error between estimated and actual phase becomes larger as the frequency increase. In this case when the frequency is equal to $2\,Hz$, the 2-sample delay represent a marginal phase shift error of $0.72\,deg$. However, when the frequency is increased to $50\,Hz$, the 2-sample delay introduce a phase shift error of $9\,deg$. It is possible to compensate for this phase shift in the software. However, phase compensation has not been discussed in this thesis and therefore not implemented.

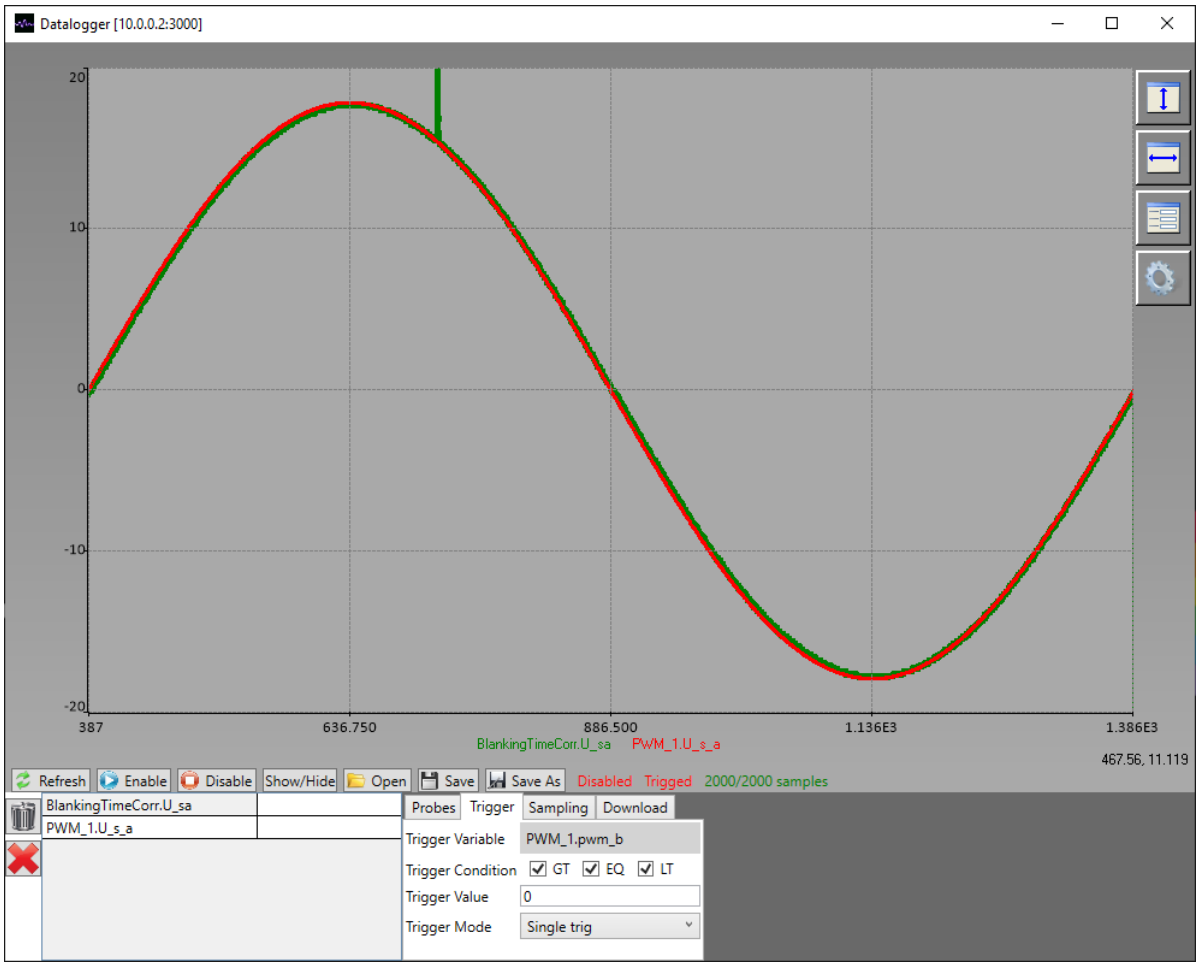


Figure 7.8: Control board – Voltage estimation without blanking time

Figure 7.9 shows the test results when blanking time is specified to be 2 μs. Again, reference voltage is shown by the red signal and estimated phase voltage is shown by the green signal. The same unilinear voltage drop that was observed during simulations is also observed during the control board test. Similarities between simulated and tested result shows that the developed software is able to estimate the distorted voltage. The result is also in accordance with the theoretical study that describes the non-linear voltage drop. It is therefore concluded that the developed software platform is able to estimate the correct output voltage given the assumptions and simplifications done in the thesis.

As mentioned in chapter 6.3, load current is emulated in the CPU and sent to the IP-core. Emulated load current is in phase with the control voltage. Since the six-step trapezoidal distortion voltage is shifted 180 deg in relation to phase current, it is expected that the output phase voltage will have a discontinuous drop in value for every 60 deg starting from the zero crossing. Looking at the result, it is evident that the estimated voltage shows this discontinuous pattern.

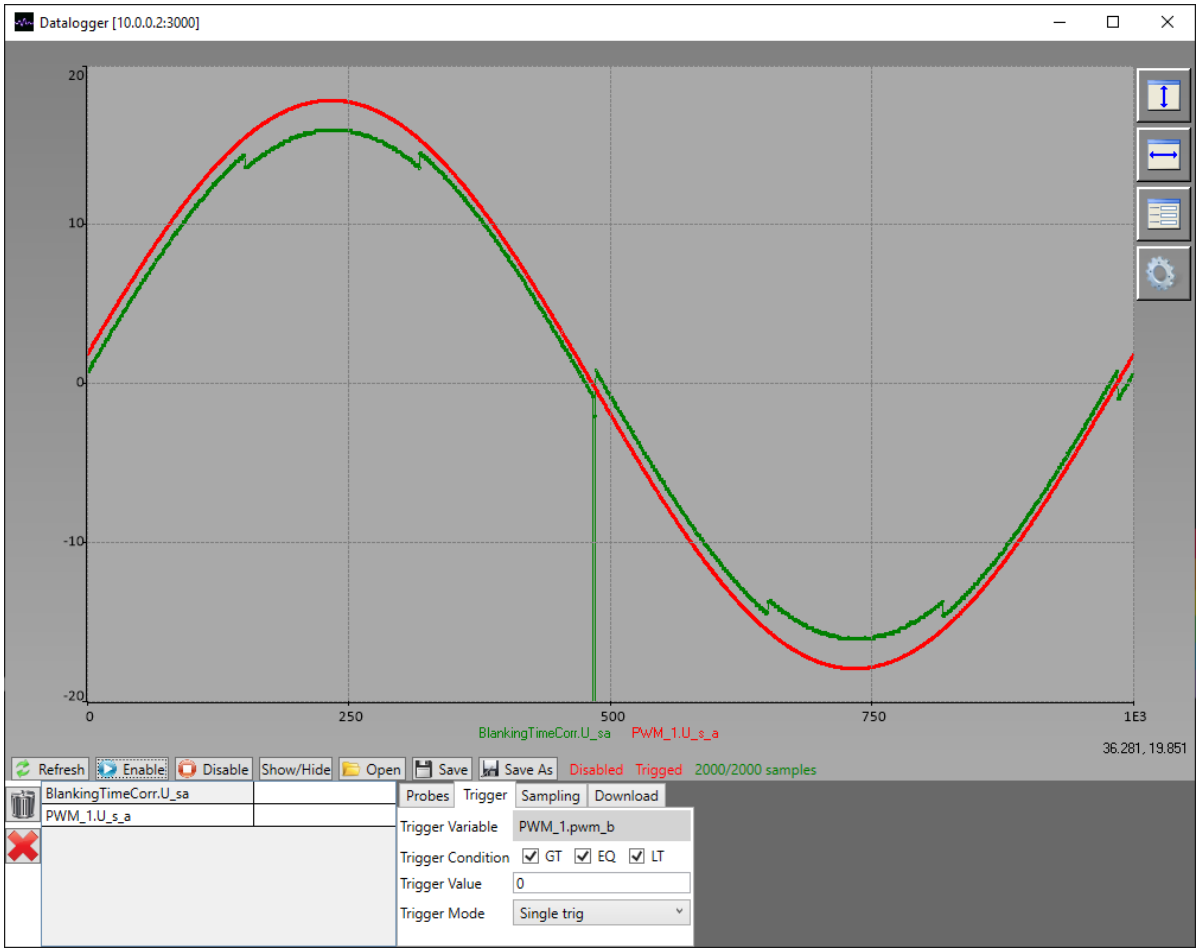


Figure 7.9: Control board – Voltage estimation with blanking time

7.4 AC-voltage test with correction

Until now, simulations and tests have shown that the developed software presented in chapter 6 are able to estimate an output voltage using the corrected on-times of the different IGBT bits, sampling tine, knowledge of current direction and DC-link voltage. The estimated voltage has shown similar waveform as was expected from theoretical analysis deduced in chapter 4.1 and shown in chapter 4.2. These chapters also show a mathematical description of the voltage drops that causes the distorted output voltage. It can be understood that this voltage is not a physical voltage drop over a component, but rather a consequence of the non-ideal properties of the switch and diode used in the inverter. It was discussed in chapter 4.2 that a reference voltage with a sinusoidal waveform would cause the output voltage to be distorted. Consequently, this distortion should be removed from the output voltage if the distorted voltage is introduced with opposite sign in the control voltage.

This chapter will present test results preformed on the control board with correction of the control voltage. As for the AC-voltage test, V/f-controller is specified to run on 2 Hz and blanking time is set to be 2 μs.

Control voltage with- and without compensation is presented in Figure 7.10 and illustrated by the blue and green signal, respectively. The distorted voltage is shown by the red signal. All signals have been scaled to pu values in relation to the DC-link voltage.

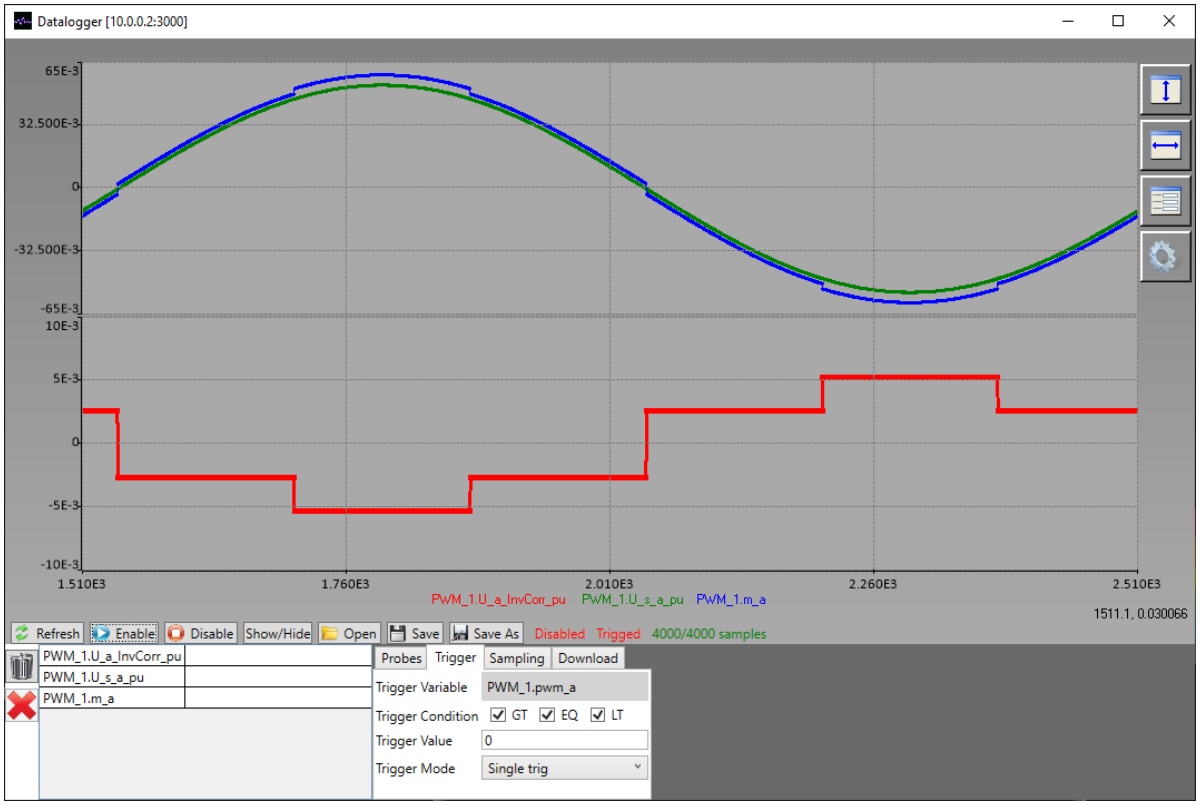


Figure 7.10: Control board – Control voltage compensation

Figure 7.11 shows the output voltage with compensation on control voltage. Estimated output voltage is shown by the green signal, modulator output is shown by the red signal and control voltage is shown by the blue signal. It is clear from the figure that the compensated control voltage removes the non-linear voltage drop from the output voltage, making the estimated voltage equal to its reference. Also, it can be observed that the estimated voltage has some irregular sampling points. This is caused by the one sampling delay introduced between CPU and FPGA. If stator currents used in the program were measured on a physical drive and not emulated in the CPU, it is expected that the compensation would use more than one sample to restore output voltage to its desired value.

There are two important aspects of this results. First, removal of distorted output voltage means that vibrations in the machine will reduce during low speed operation. Second and most important, the fact that compensated control voltage manages to remove the distortion, increases confidence in the equations deducted in the inverter model. Since voltage estimation is based on these equations, confidence in the voltage estimate is also increased.

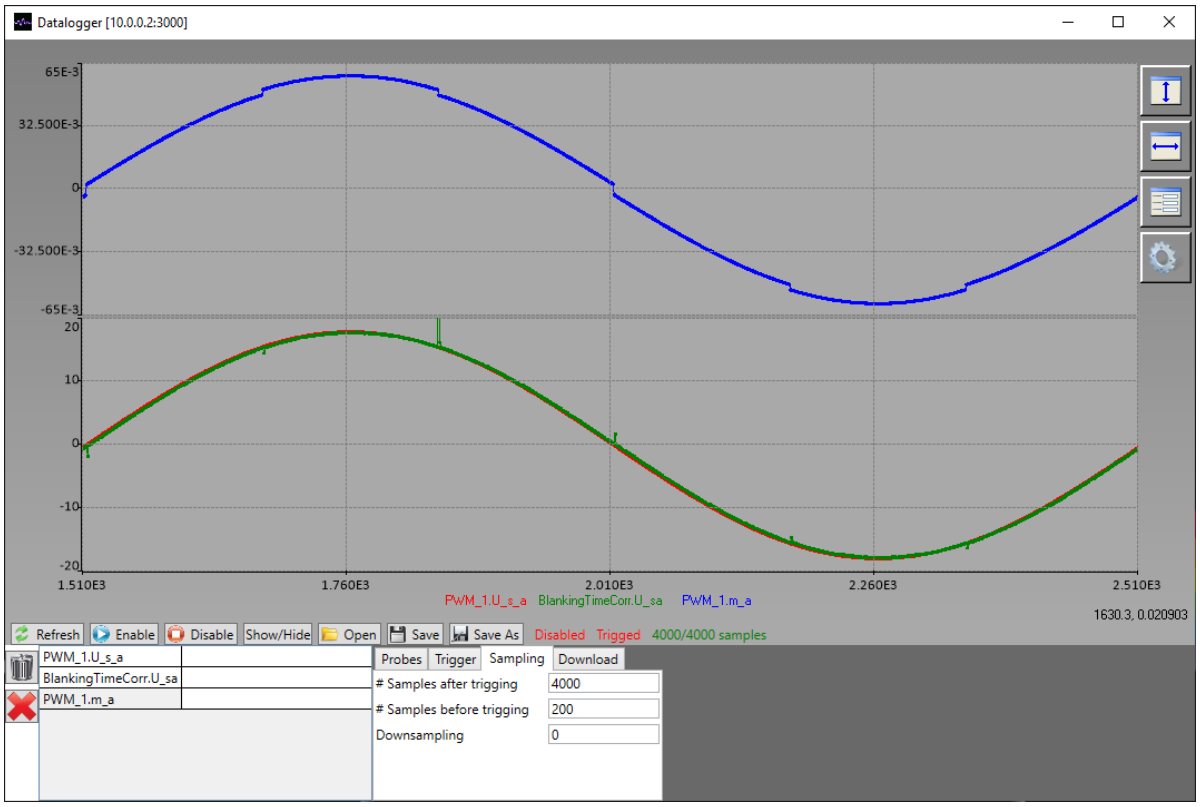


Figure 7.11: Control board – Voltage estimation with compensated control voltage

8 Discussion

In today's market, there exists several control strategies for induction machines where different design is suitable for different application areas. The control strategies can be divided into low-performance drives and high-performance drives. The PESC group at the department of Electric Power Electronic wishes to conduct research and development in the field of high-performance drives. However, since the group is establishing a new software and hardware platform, the initial focus has been on developing control routines for the CPU and signal processing in the FPGA. This has been done using a V/f-controller. V/f-control can be controlled using an open loop configuration, thus making measurements only necessary for protection. A running machine allows measurement circuits and estimation technique to be properly tested before closing the control loop and stepping into the field of high-performance drives.

The PESC group will focus on the well-known high-performance control strategy called *field-oriented control* (FOC) where the machine is controlled using an estimated rotor-flux linkage vector. Voltage model is used when the machine is driven without a position sensor, also known as sensorless control. Position sensor and wiring between sensor and controller represent a source of failure and unnecessary cost, thus making sensorless control desirable. Voltage model estimates stator flux linkage by integrating the induced voltage, i.e. difference between stator voltage and resistive voltage drop, with an open integrator. Consequently, voltage model is dependent on stator voltage, stator current and stator resistance in order to calculate flux linkages in the machine.

The open integrator used in the voltage model is sensitive to input parameter errors. Due to lack of negative feedback, inaccuracies will cause integration of a DC-offset in the flux estimation. In the literature this phenomenon is referred to as drifting. Drifting can occur during both dynamic and steady state operation and has strongest influence on the system during low speed operation. Drifting can cause vibrations in the machine and make the control unstable. Many authors have proposed different methods or so-called observers to take on this problem [4, 5, 8].

Errors in current measurement causes both torque ripples and drifting and can generally be divided into offset errors and scaling errors. Current measurements with offset errors cause the origin of the measured current vector to move away from zero while keeping a circular trajectory. The introduction of a DC-component causes the flux estimation to drift during both dynamic and steady state operation. It was also found that offset errors generated a ripple in torque with frequency equal to electrical frequency. Analysis done on scaling errors showed that the trajectory of measured current vector became elliptic. The elliptic trajectory causes drifting in stator flux estimation during dynamic operation and a wrongly estimated amplitude during steady state operation. The elliptic trajectory also causes a ripple in torque with twice the electrical frequency.

Along with current measurement errors, voltage measurement will also cause drifting in the voltage model if not done precisely. Output voltage can be measured directly on inverter terminals. However, the high rise-time and amplitude of the voltage pulses sent from the inverter requires special measuring equipment with adequate bandwidth and insulation [10]. An alternative to voltage measurement is to use the control voltage in the flux model. This is a commonly used method that gives good results in the high and medium speed range. However, during very low speed operation, voltage drops enforced by the inverter causes a non-linear relationship between output voltage and control voltage, making it unsuitable for flux estimation. Possible solutions to the problem is to use the mathematical description of the non-linear voltage drop as a feedforward term, making the control voltage sinusoidal [19, 24] or to subtract the same voltage from the distorted control voltage before using it in the flux model [10].

A third way to acquire knowledge of output voltage of the inverter is to count the time interval where bridge leg midpoint has the electrical potential of DC-link voltage. Blanking time and the difference in on- and off-time introduce time-intervals where the midpoint voltage potential is clamped to either zero or to the DC-link depending on the current direction. Accuracy of estimation therefore relay on the knowledge of these time delays. In this thesis, blanking time is considered as a constant value. This is true for inverters with a static gate driving circuit. Furthermore, the thesis also assume that on-time and off-time delays are equal in length and remain constant. However, in real-life applications the difference between the two time delays are large and can be in the hundreds of nanoseconds range [20]. Moreover, from analysis done on the IGBTs switching characteristics, it is evident that they are dependent on load current and environment.

Experimental tests conducted in [21] shows that $t_{d(on)}$ for the IGBT decreases with increasing temperatures, thus reducing the turn-on delay. Switching characteristics also show that turn-on delay is a function of current due to t_{ri}'s current dependency. Turn-off delay is dependent on the so-called Miller plateau, witch in terms is dependent on both load current and temperature [21]. The dynamic behaviour of both turn-on and turn-off delays indicates a need for current and temperature dependent expressions of these time delays in order to increase accuracy of voltage estimate.

From the discussion above, assumption of equal and constant turn-on and turn-off time may seem unreasonable. However, since tests has been conducted on the control board without a physical inverter the effect of current and temperature dependency has not been present during the tests. Furthermore, the difference in turn-on and turn-off delays, affects the estimation in a similar manner as blanking time. Therefore, by choosing a blanking time in the higher expectancy range, the estimation technique can be verified by only using blanking time. The time delays introduced by the switching characteristics of the inverter is independent of control strategy. Therefore, it can be understood that voltage estimation conducted on an open loop V/f-control, would give similar estimation results on a closed loop FOC.

Since the initial control signal for the IGBTs is generated in the PWM IP-core within the FPGA, it is obvious that corrections to this signal also has to be done within the FPGA. A blanking time correction IP-core has been designed and tested using the tools described in chapter 5. Previously, development of FPGA platform required specialist knowledge of VHDL programming. However, the ability to design and test the new IP-core in a familiar and easy programming environment as Simulink, enables many electrical engineers to participate in the development. The Simulink development platform also allows for thorough testing of the IP-cores functionality before implementing it to the design. This has reduced the debugging work substantially. Moreover, the tool auto-generates documentation of the IP-core alongside with a .h file that includes standardised initialization, read and write functions. This allows easy communication between CPU and IP-core and eliminates potential errors with respect to memory location mismatches.

To verify that the blanking time correction IP-core is able to introduce the correct time delays on IGBTs control signal, a DC-test was conducted on one of the bridge legs. Test results without blanking time showed an alternating deviation in the on-time count of 126 samples. The test results also showed that the expected on-time count was located in the middle of the two counted values. This behaviour indicates the presence of time delay somewhere within the FPGA design. The constant alternating behaviour indicate that the time delay is independent of current direction. Result when blanking time was introduced showed that the IP-core was able to reduce and extend the time duration of the IGBT control signal according to the theory. The same alternating deviation found without blanking time was still present in the on-time count. This indicate that the time delay is enforced to the IGBT control signal somewhere outside the new IP-core in the FPGA design.

AC-voltage test showed a correct voltage estimate with regard to the control voltage without blanking time. The result also showed a small phase shift between estimated and control voltage. This phase shift becomes more predominant as the speed of the machine increases, hence phase shift compensation might be needed if the estimated voltage is to be used in the voltage model. Furthermore, all AC-voltage tests showed the presence of a glitch in the estimation. The glitch is shown as a spike in estimation and was only present in the estimation during low speed operation. Investigation of the problem showed that the on-time counted value from the IP-core, sometimes was read to the CPU as zero. It was thought that internal rooting delays in the FPGA caused the on-time counter to reset before the value was latched into the register. As a solution, a delay of 10 clock cycles was introduced to the counter reset. This reduced the glitches substantially, however not all were removed.

Another observation was that the glitch only appeared when one of the currents changed direction. After reviewing the IP-core design it was found that sign of current detection logics was specified to "greater than" and "smaller than" zero. Consequently, if the current is equal to zero, both logical pathways will also be equal to zero. The AND gates seen in Figure 6.11-(a) would block all signals from passing through to the counter enable. Moreover, since the current signal is emulated in the CPU, current values in FPGA are updated with each interrupt. If the current is read in as zero, it can be understood that this might cause a sampling period with zero count.

AC-voltage tests with blanking time showed estimation of a discontinuous output voltage with a reduction in amplitude compared to the desired control voltage. This is in accordance with the theoretical study, where it was explained that a sinusoidal control voltage would cause a distorted output voltage. Analysis showed that output distortion could be removed if the inverse of the non-linear voltage drop was added to the control voltage. An expression for non-linear voltage drops was deduced in the non-ideal inverter model. Test results showed that the corrected control voltage was able compensate for the voltage drop, thus making the output voltage equal to the reference value calculated in the modulator.

There are several reasons why this result is important. During open loop control, distorted output voltage would create unwanted vibrations in the machine during low speed operation. Vibrations in electrical machines reduces lifetime of mechanical components like bearings and seals and is generally unwanted, thus making control voltage compensation desirable. Moreover, the fact that the distorted output voltage is restored back to the reference value, validates the equations deduced in the non-ideal inverter model and the relationship between control voltage, non-linear voltage drop and output voltage equation. It also shows that equations and estimation technique are implemented correctly on the control board. Lastly, the open loop control voltage with correction has the same waveform as a closed loop controller. Since the estimated voltage is equal to its sinusoidal reference, it is reasonable to believe the estimation technique can be transferred to closed loop control without having to do any adjustments. Again, this shows that the estimation technique can be utilized on both V/f-control and FOC.

It is important to emphasise that the results presented in this thesis are performed purely on the control board without a real inverter or load connected. It is expected that uncertainties in delay parameters and real current measurement would cause a reduction in the accuracy of voltage estimation. This might call for tuning the estimator once it is connected to a physical drive. However, the results clearly show that the voltage estimator works in an ideal environment, proving the concept and the mathematical relations deducted in the thesis.

9 Conclusion and Further Work

9.1 Conclusion

This master thesis focuses on modelling and control of an induction machine drive. It also focuses on implementation of an on-line voltage estimation technique for a two-level three-phase inverter. The on-line voltage estimator has been developed using Xilinx SDK, Xilinx Vivado and Xilinx System Generator DSP tool in MATLAB Simulink. The programming platform used in this thesis will become a part of a common programming platform at the department of Electrical Engineering at NTNU. Therefore, programming structures and tools used in the thesis is also thoroughly explained.

Sensorless control of induction machines is desirable due to reduced cost and increased reliability. In FOC, voltage model is used as flux estimation model during sensorless operation. However, errors in measurements and estimation causes drifting in the estimated reference frame due to the open integrator. Drifting introduces unwanted vibrations in the machine and can even cause the control to become unstable. The impact of drifting is amplified during low speed operation and slow runs though zero speed.

Being one of the contribution factors to drifting, it is important that voltage measurements or estimations are accurate. From the non-ideal inverter model, it was seen that output voltage of the inverter was given by the midpoint potential of the inverter bridge legs. Analysis showed that blanking time and the difference in turn-off and turn-on times of the IGBTs introduced a time delay to the voltage potential, compared to the IGBTs control signal. It also showed that the time delay was current dependent.

The voltage estimator's ability to correct time duration where midpoint potential of the bridge leg was equal DC-link voltage was initially tested. The tests results without blanking showed an alternating deviation between the on-time counts of 126 samples. This deviation was found to be independent of current direction, and the nature of the deviation suggested that the IGBT control signal was subjected to a delay somewhere in the FPGA design. Test results with blanking time showed the desired delay in on-time of every other count. Measurements showed that the observed delay was equal to blanking time. It also showed that the delay was current dependent. The same alternating deviation found without blanking time, was also present when blanking time was introduced.

The voltage estimator was also tested during low speed V/f-control. Tests with and without blanking showed that the voltage estimator was able to perform satisfactorily. Estimated output voltage followed control voltage without blanking time and the expected non-linear voltage drop was observed when blanking time was introduced. Furthermore, when compensation for non-linear voltage drop was introduced in the control voltage, results showed that output voltage followed reference voltage from the modulator.

The tests were conducted on the control board with no inverter connected. It is expected that test results of voltage estimation when connected to a physical drive, would show a lower accuracy than presented in this thesis and tuning of the estimator might be needed. However, given the test results it can be concluded that the voltage estimator was able to estimate correct output voltage given the correct input parameters.

Finally, time duration where midpoint potential is equal to DC-link voltage was estimated in the FPGA design using a self-developed IP-core. The IP-core was designed and tested using Xilinx System Generator tool in MATLAB Simulink. This tool has proven to be an effective way of programming complex FPGA programs without knowledge in VHDL programming. The tool has the potential to enable electric power engineers the ability to participate in designing, building and testing advanced FPGA designs, something that previously has required specialist programming knowledge.

9.2 Further Work

The goal of the theoretical study, programming tools, programming structures and developed control software presented in this thesis, was to contribute in building a common programming platform. Once the programming platform is established it will allow for more research and development in regard to high-performance sensorless FOC.

Following tasks could be considered for further work in the development to the programming platform:

- Investigate effects in voltage estimation accuracy by making duration of turn-on and turn-off delays dependent on load current and temperature.
- Improvements to the blanking time correction IP-core to further reduce or remove glitching in voltage estimation.
- Experimental testing and tuning of the voltage estimator using a physical motor drive setup.
- Design and develop a current measurement circuit for protection and measurement.

References

[1] W. Zimmermann, "Sensorless Control of AC Machines," Fachhochschule fuer Tecknik Esslingen, Available: http://www.it.hs-esslingen.de/~zimmerma/publications/erk.pdf.

[2] M. Leksell, L. Harnefors, and H.-P. Nee, "Machine Design Considerations for Sensorless Control of PM Motors," KTH / Royal Institute of Technology, Stocholm1998

[3] F.Bauer and H. D. Heining, "Quick Responce Space Vector Control for a High Power Three-level-Inverter Drive System," *EPE*, pp. 417-421, 1989.

[4] M. C. Paicu, I. Boldea, G.D.Andreescu, and F.Blaabjerg, "Very low speed preformance of active flux based sensorless control: interior permanent magnet synchronous motor vector control versus direct torque and flux control," *IET Electr. Power Appl.* , vol. vol. 3, no. no. 6, pp. 551-561, 2008.

[5] M. Niemelä, "Position sensorless electrically excited synchronous motor drive for industrial use based on direct flux linkage and torque contro," Doctoral thesis, Lappeenranta University of Technology, 1999.

[6] M. Bolstad, "Sensorless Control of Synchronous Machines Used in Adjustable Speed Hydro," Department of Electric Power Engineering, NTNU, Trondheim, 2018.

[7] E. Mørkved, "Sensorless Control of a 6-phase Induction Machine," MSc, Department of Electric Power Engineering, NTNU, Trondheim, 2018.

[8] T. F. Nestli, "Modelling and identification of Induction Machines," Doctor Degree, Department of Electric Power Engineering, NTH, Trondheim, 1995.

[9] V. Fjellanger, "Sensorless Control of a six-phase Induction Machine," Department of Electric Power Engineering, NTNU, Trondheim, 2018.

[10] Joachim Holtz and J. Quan, "Drift- and Parameter-Compensated Flux Estimator for Persistent Zero-Stator-Frequency Operation of Sensorless-Controlled Induction Motors," *IEEE Transactions on power electronics*, vol. vol. 39, no. no. 4, pp. 1052-1060, 2003.

[11] R. Nilsen, *Electric Drives*. Trondheim: Department of Electric Power Engineering, 2018.

[12] T. Wildi, "Electrical Machines, Drives, and Power Systems," 5. ed. Upper Saddle River, New Jersey Columbus, Ohio: Prentice Hall, 2002.

[13] N. B. Araya, "Modelling and Control of Six-Phase Induction Motor Drive," MSc, Electric Power Engineering, NTNU, Trondheim, 2012.

[14] S. Midttveit and R. Nilsen, "Transformerte Modeller for Driefeltmaskiner," EFI SINTEF-Gruppen03.06.1987 1987.

[15] A. M. Trzynadlowski, *The Field Orientation Principle in Control of Induction Motors* Reno: Kluwer Academic Publishers, 1994.

[16] A. Hughes, *Electrical Motors and Drives*, 2ed ed. Manchester: G H Newnes, 1993.

[17] B. Fossen, "Modelling and Identification of Multi-phase Machines," Project thesis, Department og Electric Power Engineering, NTNU, Trondheim, 2016.

[18] Dae-Woong Chung and S.-K. Sul, "Analysis and Compensation of Current Measurement Error in Vector-Controlled AC Motor Drives," *IEEE Transactions on industry applications*, vol. vol. 34, no. no. 2, pp. 340-345, 1998.

[19] J. Holtz and J. Quan, "Sensorless Vector Control of Induction Motors at Very Low Speed Using a Nonlinear Inverter Model and Parameter Identification," *IEEE Transactions on industry applications,* vol. vol. 38, no. nr. 4, pp. p. 1087 - 1095, 2002.

[20] J.-W. C. S.-K. Sul, "Inverter Output Voltage Synthesis Using Novel Dead Time Compensation," *IEEE Transactions on power electronics,* vol. vol. 11, no. no. 2, March 1996.

[21] H. L. Hove, O. C. Spro, D. Pefritsis, G. Guidi, and K. Ljøkelsøy, "Minimization of det time effect on bridge converter output voltage quality by use of advanced gate drivers," *2019 10th International Conference on Power Electronics (ICPE 2019 ECCE Asia),* 2019.

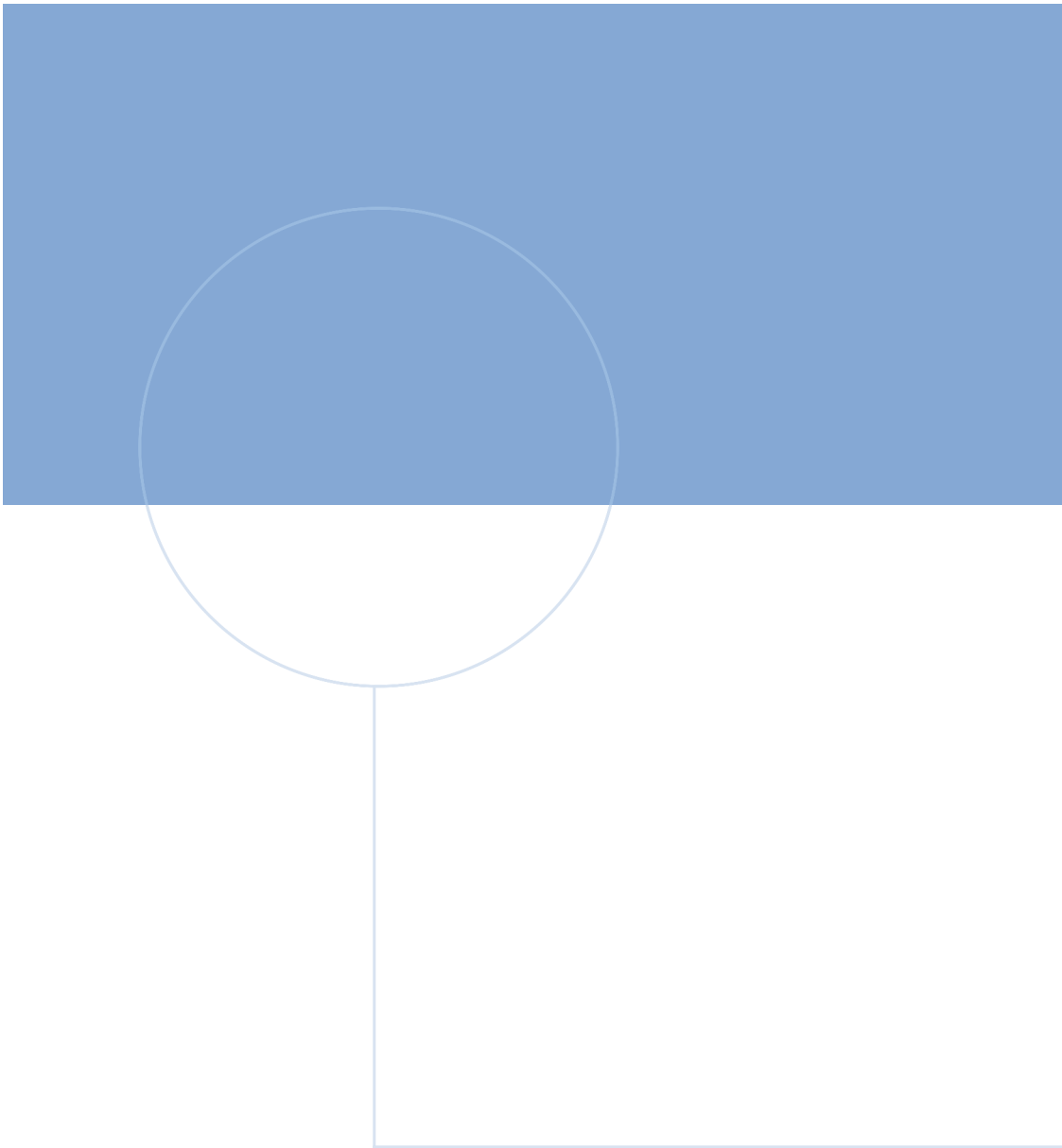
[22] N. Mohan, T. Undeland, and W. Robbins, 3., Ed. *Power Electronics*. USA: Wiley, 2003.

[23] J.-W. Choi and S.-K. Sul, "Inverter Output Voltage Synthesis Using Novel Dead Time Compensation," *IEEE Transactions on power electronics,* vol. vol. 11, no. no. 2, March 1996.

[24] G. Pellegrino, P. Guglielmi, E. Armando, and I. R. Bojoi, "Self-Commissioning Algorithm for Inverter Nonlinearity Compensation in Sensorless Induction Motor Drives," *IEEE Transactions on Industry Applications* vol. vol. 46, no. nr. 4, pp. p. 1416 - 1424, 2010.

[25] P. Horowitz and W. Hill, Third, Ed. *The Art of Electronics*. New York: Cambrige University Press, 2015.

[26] C. Visjon. (2018). *Zynq All Programmable SoC System Architecture* [Online]. Available: https://www.core-vision.nl/events/zynq-all-programmable-soc-system-architecture-8/?lang=en



NTNU
Norwegian University of
Science and Technology